September/October 1980

Single Copy Price $7.50

I/O News

Volume One, Number One

THE CROMEMCO STORY

INTRO TO COMPUTER GRAPHICS

THE CDOS ACTIVE COMMAND FILE

The OFFICIAL PUBLICATION OF THE INTERNATIONAL ASSOCIATION OF CROMEMCO USERS

Huge Floppy Disc Storage... That's What Cromemco's New Controller Card Offers

Present computers can be upgraded

Since early September all Cromemco computer systems have been manufactured with two to four times the previous (and already substantial) floppy disk storage capacity.

That means Cromemco's floppy disk capacity is increased to as much as 4.8 megabytes.

Such a truly remarkable capacity is now available in Cromemco's System Three with four disk drives using 8-inch diskettes.

The large capacity results from Cromemco's use of double-sided disk drives and an exciting new disk controller card that not only permits double-density storage but has other delightful features such as self-test of your system.

This all means that Cromemco leads the industry by being the only manufacturer to offer customers a choice of quad-capacity 5-inch or quad-capacity 8-inch disk drives in its various systems, the System Three, System Two, and Z-2H hard disk/floppy disk system.

NEW RDOS-2 WITH NEW FEATURES

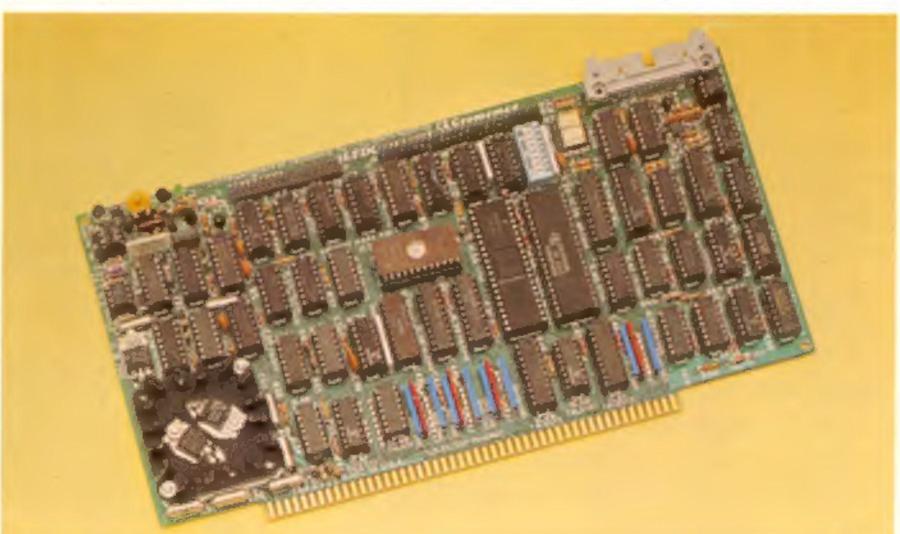
The new Model 16FDC controller card includes a 4K byte ROM on board with a new and greatly expanded resident disk operating system called RDOS-2.

The first new feature you'll find in RDOS-2 is that even if your system is set to automatically boot in CDOS on power up, you will still be able to abort the boot in order to enter the RDOS-2 command mode simply by depressing the ESCAPE key on your terminal during the boot sequence. No switch on the 16FDC needs to be changed in order to do this.

Once in the RDOS-2 command mode you have the capability to

Continued on page 13

Versatile new 16FDC controller card brings double-sided, double-density operation and resident self-test diagnostics to all Cromemco systems.



Breakthrough for the 80's:

LISP Comes of Age and Cromemco Hands You the Keys to the Door

By: Lois Flynne, Ph.D.
Professor of Information Science,
San Francisco State University

This year LISP celebrates its twenty-first birthday, a non-pareil among programming languages. Unchallenged in its role as the "calculus" of the Artificial Intelligence field, LISP has been used to tackle some of the most difficult of programming tasks. Yet, rumors to the contrary, LISP is perhaps the friendliest of programming languages, more highly conversational than Basic though considerably more sophisticated and powerful in terms of the tasks it can readily accomplish.

Continued on page 14

The Cromix* Operating System for Cromemco Computer Systems

By: D. Thompson McCalmont
Software Engineer, Cromemco, Inc.,
Mountain View, CA 94043

An operating system is a program which supervises and manages the execution of other programs in the computer. The functions of an operating system include: performing all types of I/O, including I/O to mass storage devices such as disks and tape

Continued on page 23

*Cromix is a trademark of Cromemco, Inc.

table of contents

September/October 1980 Volume I, Number One

I/O News The Official Publication of The International Association of Cromemco Users is a bi-monthly magazine available to members of the association only. Its editorial and advertising policies are designed for the enlightenment of its readers in regard to new uses for, and developments of, Cromemco products and other products compatible with Cromemco equipment and systems.

COVER FEATURES

ARTICLES & FEATURES

COMING SOON

More on Cromix
Evaluation of a Forms Package
Evaluation of Complex Numbers on FORTRAN IV
Another Article by Darwin Engwer
Update on Color Graphics
And much, much more

I/O News (ISSN # pending) is published bi-monthly by the International Association of Cromemco Users (a California corporation), P.O. Box 17658, Irvine, CA 92713. General Offices are in Newport Beach, CA and can be reached by telephone at: (714) 955-0432. Subscriptions to I/O News are not sold separately. Members receive I/O News only in conjunction with their membership in the IACU.

Controlled Circulation Postage paid at Irvine, CA 92713. POSTMASTER: Send address changes to I/O News, P.O. Box 17658, Irvine, CA 92713.

Return postage must be included with all manuscripts and photos submitted if they are to be returned. The IACU and I/O News accept no responsibility for the return of unsolicited materials.
All rights in letters sent to IACU and I/O News will be treated as unconditionaly assigned for publication and copyright to comment editorially and to edit.

Richard Kaye
Editor and Publisher

Pascal/MT® +
The Choice of Professionals, Worldwide

Pascal/MT+ is an enhanced version of our successful Pascal/MT package popular with both hardware and software engineers. Generating even more optimized ROMable 8080/Z80 object code (directly from the Pascal source), Pascal/MT+ now includes those features demanded by today's professional. We've added modular compilation and linking (incorporating the industry standard linker format used by FORTRAN and PL/I) and user selectable Z80 native code or our enhanced 8080/Z80 object code. Also included is an improved SYMBOLIC DEBUG, more I/O facilities and even an option to produce pseudo assembly output intermixed with the Pascal source.

MODULAR COMPILATION

Pascal/MT+ generates the same industry standard relocatable code used in FORTRAN and PL/I. Pascal code may be linked with our linker or any compatible linker. Both Pascal and assembly language modules may be separately compiled and then combined to produce a final program. With modular compilation available, the run time overhead becomes as small as 256 bytes, and is typically 1200 bytes.

OPTIMIZED NATIVE OBJECT CODE

Pascal/MT+ now produces even more optimized object code than before. Now included is a compile-time selectable option to generate Z80 code where it is more efficient. The 8080/8085 code is better, too.

IMPROVED DEBUGGING FACILITIES

The Pascal/MT+ debugger is improved. We've added facilities to use the debugger with larger programs and even in a ROM-based environment. The compiler can also output a disassembled listing of your programs with the Pascal source code interspersed between the object code. This helps you gain visibility in non-CP/M* applications.

ENHANCED I/O CAPABILITIES

To our already powerful console, printer and re-directable I/O facilities we have added Pascal standard TEXT file I/O as well as improved line editing for console I/O.

PLUS . . .

- Transcendental functions
- Improved String Handling
- Benchmark tests rate our code up to 10 times faster than P-code Pascal and faster than other native code high level languages too!
- Re-assembly of the run-time package no longer required for ROM based applications!

AND OUR STANDARD FEATURES:

- High speed compilation (up to 2000 lines per minute)
- Non-CP/M* environments supported
- CP/M* file support
- Program chaining
- Built-in mini-assembler
- Compile-time constants
- Business (18-digit) arithmetic
- Scientific (6.5 digit) arithmetic
- AMD9511 Hardware support
- Source code for run-time package included (re-assembly now requires Macro-80 assembler)
- I/O port, Interrupt facilities
- Bit and Byte manipulation
- Many useful built-in procedures (loaded from run-time library)

SYSTEM REQUIREMENTS

To execute the Pascal/MT+ system you must have a CP/M* system with at least 32K of memory. For larger applications we recommend 48K or more. Source programs are prepared using ED (or equivalent), we recommend WORD-MASTER.

(Coming soon: versions for Intel ISIS-II and Heath HDOS. Ask us.)

ORDERING

STILL ONLY $250 will buy you one of the best software power tools available.

Pascal/MT+ has been chosen by companies such as GE, RMG and Chromatics (as well as over 700 other companies) as the Pascal for them, why not join them!

The Pascal/MT+ package:

- BCD compiler
- Floating point compiler
- Linker
- Interactive Symbolic Debugger
- Run time package in Source and Object form.
- Pascal library utility routines
- 95+ page user's guide
- Sample programs

The user's guide is available for $30.00, refundable with a system purchase.

MT microSYSTEMS has a very reasonable graduated, one time royalty arrangement for free standing software generated by Pascal/MT. Ask us.

To order Pascal/MT+ write or call:

MT MicroSYSTEMS

1562 Kings Cross Drive
Cardiff, CA 92007

(714) 753-4856

We ship on 8" single density and 5-1/4" North Star single density. Other formats available. Ask us.

CP/M is a trademark of Digital Research
Pascal/MT is a trademark of MT microSYSTEMS

Who are those guys, anyway?

THE CROMEMCO STORY

Cromemco, Incorporated. The largest manufacturer of general purpose microcomputers in the world. Some 350 employees, 200,000 square feet of production facilities in the very heart of Silicon Valley, dozens of products shipped worldwide to thousands of customers, producing millions of dollars in annual sales.

Cromemco, Incorporated. Home of some of the most creative minds ever assembled in one place since the space program was in its heyday.

There are many stories as to how Cromemco achieved such prominence in less than ten years.

Some of them start, "He was only 19 when his father died, leaving him a '31 Model-A Ford, a tool company, and a million bucks, but Howard Hughes somehow managed..."

That's not the Cromemco story. Actually, Cromemco's story has never before been written. Maybe that's because it's not the modern 'riches-to-riches' story like the

Hughes legend. It's more of a Horatio Alger story that started with a $300 royalty check earned in 1971 by two Ph.D. candidates at Stanford University.

The two graduate students, Harry Garland, fresh out of Kalamazoo College in Michigan with a brand new degree in mathematics, and Roger Melen, motoring to Palo Alto with the ink still wet on his Electrical Engineering[1] degree from Chico State College in Northern California, both arrived at Stanford's campus in September of 1968. And both were assigned to live at Crothers Memorial Hall, the graduate student dormitory.

Many people believe in luck, chance, coincidence, or whatever name applies to events which cannot be logically explained. There may be a few elements out of the Twilight Zone in the Cromemco story. Who can say for certain? But, there is more evidence of a chain of events with the links forged by the strenuous demands of the Stanford Graduate Program.

Of those few applicants from around the world actually admitted, fewer than 25% attain the coveted Ph.D. degree, and most of the attrition occurs within the first year. Those who survive the rigors of the program do so through a combination of native intelligence, hard work, and nerves of steel. Melen and Garland spent their first year at Crothers Memorial Hall learning to survive.

Writing for Fun and Profit

But, the characteristics that brought them to Stanford, and through their first year, as individuals tended to bring them into closer contact as they climbed the graduate student ladder. Each had both a personal and a professional interest in electronics, both were amateur radio operators,[2] and both had cut their computer teeth on IBM's 1620 which boasted 20K of memory, a card reader, and a 2-megabyte hard disk.

As ham radio operators, both Garland and Melen were active in home-brewing various radio circuits. One of Melen's circuits for an active audio filter was published in Wayne Green's 73 Magazine.[3] His article appeared in 1969 under the title, "Build the Beatnote Basher, A Selective Audio Filter."

Success breeds success, and with the appearance of the Beatnote Basher article, Garland and Melen determined to collaborate on other articles in the electronics field. Their timing could not have been better. The embryonic electronics industry which had just enabled Neil Armstrong to take his historic "...giant leap for mankind..." was making quantum leaps in integrated circuit technology, and the first commercial microprocessors were on the drawing boards.

There was no doubt that Garland and Melen had selected the right area of endeavor. Their articles were in instant demand. On October 28, 1970 they received a royalty check for $300 from Popular Electronics magazine — the first income ever received by the Garland/Melen partnership — but not the last. In 1971 alone, three more of their articles appeared in Popular Electronics,[4] and they published their first book, Understanding IC Operational Amplifiers.

The Mystique of Crothers

Throughout this period, the prolific duo were deeply involved in their graduate studies which brought them into constant contact with others immersed in the Stanford academic community. A few were to play important roles in an as yet undreamed of company.

One of these, an energetic Electrical Engineering student (and 1972 Olympic Medalist with U.S. swim team) approached Melen with a question on operational amplifier design. Melen, the co-author of a book on that very subject, was the right person to ask — and the right person to know. The student, Brian Job, went on to receive his Electrical Engineering degree from Stanford, and following his Olympic success, earned a Master's degree from the Harvard Business School. Job's encounter with Melen led him to join a fledgling Cromemco as its first Sales Manager — a position he holds today.

Back at the dorm, Alice Ahlgren, studying for her Ph.D. in communications, decided Crothers Memorial deserved its own newspaper. The Rag brought pleasure to the residents, and captured the attention of Roger Melen. (Melen reportedly waited inside the closed door of his dorm room until The Rag was slipped under his door, then immediately shot it back out into the hall to Alice Ahlgren's surprise.) Shortly after receiving her degree, Dr. Ahlgren joined Cromemco in charge of the company's wide range of publications.

Others from the graduate school days made a lasting impression. One was Harry Garland's roommate, James Patrick McVittie, who often acted as a sounding board for Garland when he was preparing his numerous articles. Today, he is Cromemco's Reliability Engineering Manager.

One of the most endearing relationships was established with a coed from a neighboring college Margaret Hogan would turn out to have an important influence on the developing company, for when she married Harry Garland, she also found her life enmeshed with Cromemco as its first personnel manager. Margaret Garland still manages Cromemco's Personnel Department.

The Post-Graduate Years

In 1972, both Garland and Melen were awarded their coveted Doctor of Philosophy degrees — Melen's in Electrical Engineering, and Garland's in Biophysics with a minor in Electrical Engineering, and both were offered the rare opportunity of continuing at Stanford as Research Associates with Dr. John Linvill. In addition to being Chairman of the Stanford Electrical Engineering Department, Dr. Linvill had gained worldwide renown as the inventor of the Optacon blind reading aid. Dr. Linvill's vision extended far beyond his technical accomplishments, as he recognized the particular synergy that existed between his two young colleagues and encouraged their continued collaboration on projects outside the research laboratory.[5]

The First Garland/Melen Product

And continue their collaboration they did. A 1973 article, "Build a Low Cost Op Amp Tester," that appeared in Popular Electronics elicited a strong response from Art Salsberg, the magazine's Editorial Director, who suggested that a kit of the parts described in the article be marketed by the authors. In June, 1973, Salsberg wrote Harry Garland:

> "We chatted with Edward Roberts, President of MITS, a very successful kit manufacturer. We described your kit to him (the OP AMP Tester) and he expressed great interest. I believe it would be to your benefit to contact him to discuss the various ramifications of getting your project into kit form."

MITS — the same company that would introduce the computer that began the microcomputer revolution — offered the Op Amp Tester as a kit, and agreed to pay Garland and Melen a royalty of 5% of sales. A small link in a complicated chain of events was beginning to take shape.

An article entitled, "A Single IC Capacitance Meter" appearing in the February, 1974 issue of Popular Electronics forged yet another link in the chain. Again, the editors suggested a kit of parts be made available. But, this time, Garland and Melen determined to control their own destinies (no royalties had yet been paid by MITS). With the introduction of their capacitance meter kit, the pair — still in their twenties — entered the manufacturing business.

Later that year, Roger Melen went to New York to discuss the details of two upcoming Popular Electronics articles with Les Solomon, the Technical Editor. Solomon, however, could barely contain his enthusiasm over an "...incredible minicomputer kit with unprecedented capability ..." that would sell for only $397. The computer, named the Altair by Solomon's daughter, was developed by Ed Roberts of MITS with a system bus that provided for tremendous system expansion, and he had submitted the material in hopes that other readers would submit projects bus compatible with the Altair.

That was enough for Melen. He diverted his return flight to Albuquerque for a meeting with Roberts. At worst, he could check on the long overdue royalties.

Roberts met Melen's plane and they proceeded to the MITS fac-

(1)Melen was honored as Outstanding Engineer by the California Society of Professional Engineers.

(2)Garland's call sign was WA8FJW. Melen's was WB6JXU.

(3)Green later founded Byte and Kilobaud magazines.

(4)See bibliography for references.

(5)His faith in his instincts is rewarded today as he oversees his two former assistants' accomplishments from his position on Cromemco's Board of Directors.

tory where they worked until five o'clock the next morning. The project that consumed their time and energy was interfacing the Cyclops, a solid state TV camera and the subject of an upcoming Garland/Melen/Walker article[6] to the Altair computer. After pouring over the Altair schematics, reviewing the bus structure, and exploring DMA interface, they concluded Cyclops interface could be accomplished. Roberts agreed to ship one of the very first Altair computers when production began.

As for the overdue royalties, they would have to wait. MITS had sunk everything into the Altair.

(From left) Dr. John Linvill, Dr. Harry Garland, Dr. Roger Melen.

The Beginning of an Era

As 1974 wound down, Garland and Melen took a reflective look at their progress. The Capacitance Meter Kit was selling very successfully at $19.95 and they were preparing to market the new Cyclops Kit. It was time to formalize the partnership, and on December 11th, "Cromemco," named after their beloved Crothers Memorial Hall, was filed as a partnership with the County Clerk of Santa Clara County, California. It was the end of an era of graduate school enthusiasm and inventiveness, but the beginning of a new, much more sophisticated period.

On February 11, 1975, Altair Computer #0002 arrived in Palo Alto, and Melen arranged for Terry Walker, an electrical engineering doctoral candidate, and co-author of the Cyclops article, to lead the design effort to interface the Cyclops camera to the Altair.[7]

They created a device which could display the low-resolution, black-and-white picture. But why stop there? They soon discovered that the display could have up to 128-by-128 point resolution, color capability, and that the interface could be used with an ordinary home TV set. And all this could be accomplished for only modest increases in cost and complexity.

The result was a dazzling color display, and Cromemco had its newest product: The Dazzler.

An Era of Accomplishment

The next eighteen months produced a level of activity and development that could only be likened to preparations for a major space shot. Much of it emanated from the Cromemco Research Lab (the second bedroom of Roger Melen's apartment) and from the Cromemco Production Facility (Harry Garland's garage).

One of the first projects was the creation of demonstration software for The Dazzler — a laborious effort as no assembly language existed. Programs had to be entered in raw machine language, and a single mistake could totally destroy hours and hours of work. Ed Hall (today a Cromemco Software Engineering specialist) was assigned the painstaking project. He came through.

The building of the demonstration program pointed out the need for Cromemco's next project: a non-volatile read-write memory board using UV-erasable PROMs. Named the Bytesaver, it quickly became the most popular add-on board ever developed for the Altair.

The Dazzler received its first major public showcase in September at WESCON in San Francisco. It was there that Les Solomon, who one year earlier had introduced Melen to the Altair computer, became the first person in the publishing world to grasp the potential of low cost computer graphics displays. His enthusiasm over what he witnessed led him to give The Dazzler a front cover splash on the February, 1976 issue of Popular Electronics.

With Cyclops selling well, and Bytesaver and The Dazzler about to be made available in volume, the Cromemco partners felt it time to move the company out of their homes. In December, 1975 they moved to a 200 square foot office in Los Altos and hired Sonia Garbarino, their first full-time employee[8].

The new office was just around the corner from another company in its infancy. Zilog was then developing a microprocessor destined to become the successor to the 8080 microprocessor used in the Altair. Cromemco became Zilog's first customer for the innovative Z-80 CPU (ZPU) card.

Cromemco and Zilog were not the only companies making giant strides in the embryonic microcomputer industry. During 1976, both Imsai and Polymorphic Systems introduced Altair compatible mainframes. And Processor Technology's 4KRA memory boards were the most popular add-on memory cards ever developed for the Altair.

(6) Popular Electronics, **February, 1975.**

(7) Terry Walker is still with Cromemco as a valued member of the technical staff.

(8) Today, Sonia Garbarino is a supervisor in the Sales Department.

The Birth of The S-100 Bus

Multi-manufacturer support for the Altair mainframe grew. And as it grew across the breadth of the burgeoning industry, reference to the machine's bus structure as the "Altair bus" became less and less appropriate. This was the topic of a conversation between Garland and Melen in August of 1976 on a TWA flight enroute to the Personal Computing exhibition in Atlantic City.

Melen perceived the impact of this bus on the microprocessor industry, noted that it needed a more universal designation, and suggested the Standard bus. Garland agreed with the concept, but observed that standards usually have numbers associated, and suggested that it be renamed the Standard-100 bus, or S-100 bus for short. The partners agreed, but horizontal industry support was needed to affect a permanent change.

Luckily, Bob Marsh, Vice President and General Manager of Processor Technology was on the same flight. The Cromemco principals woke him from his nap long enough to get his agreement to the S-100 designation.

Melen wheedled a yes vote out of Polymorphic Systems and a neutral out of Imsai at the Atlantic City convention, and the space age term — born close to the edge of space on a TWA jet — was on its way to its present acceptance. Today the S-100 bus is the most widely supported bus in the computer industry, and the only such bus formally recognized in a proposed IEEE standard.

The End of an Era

By the end of 1976, Cromemco had 15 full-time employees, (14 of whom are still with the company) a reputation as a pioneer in the new frontier of microelectronics, and a further reputation for having highest quality and greatest reliability of any company in that industry. In addition, Cromemco had its first catalog, offering a grand total of eight items:

8K Bytesaver
TV Dazzler
D + 7A Analog Interface

[illegible] mera and Interf [illegible]

[illegible]

What had started with a magazine article by two eager graduate students at Stanford's Crothers Memorial Hall had evolved into a leading business in an explosive industry. It was time for the partnership to end, and time for a new beginning.

On December 31, 1976, Articles of Incorporation were filed with the California Secretary of State.

Cromemco, Incorporated was born.

(Editor's note)

We can only speculate about the f[illegible] of Cromemco. But there are more than a few hints that indicate it is bound to be a very exc[illegible]

For example, the [illegible] four different [illegible] seven major peripherals, 20 available computer cards, and more software than we care to count. And, there is a new catalog coming out this month! Inasmuch as Cromemco has averaged one new product per month fo[illegible] he hat [illegible] es a few days before this first issue of went to press. It was a very impressive tour. But, the part of the tour that left me in a state of awe was a several thousand square foot area in one of the new buildings. I was told this area was devoted exclusively to Research and Development, and that it was built to provide facilities for one hundred people. One hundred scientists, technicians, programmers, and engineers all engaged in R & D for a company that less than four years ago had only 15 employees.

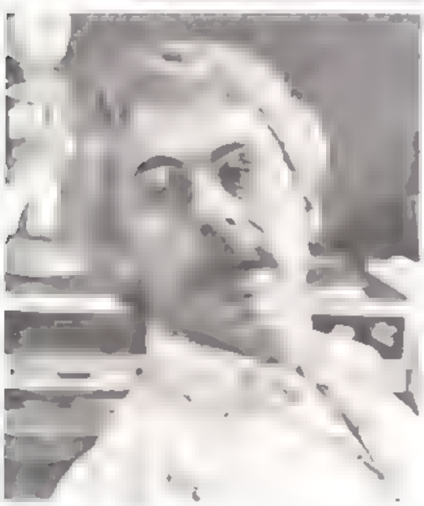
Yes. The future will be very exciting.

Have You joined IACU? See page 32.

output...

Welcome to the premiere issue of I/O News. And welcome to the IACU, perhaps the fastest growing users' group in the world. Membership at the end of August was approaching 800 — with some 30% international — and we are growing daily. The response has been greater and quicker than we anticipated. We thank you.

The purpose of the IACU is to act as a communications center — an information clearing house between Cromemco, its users, Dealers, and OEMs. We are a service organization existing solely to disseminate information about Cromemco products, processes and applications. The reason for our existence is simply this: the technological growth of microcomputing has far out-stripped the communications concerning that growth.



Although we are in independently owned California Corporation, we were organized as a result of Cromemco's outstanding success. And, we have had the total support and encouragement of Cromemco in establishing this association. We value Cromemco's support, and while we are very interested in any valid, constructive criticisms, we have no intention of developing an adversary relationship with Cromemco. Let's face it...without Cromemco's enthusiasm for our efforts, we would not be bringing you this magazine.

Nor would we be able to provide some of the other communications services that will be phased in over the next few months. Things like a computerized Bulletin Board connected to a telephone modem that will allow members to call from anywhere in the world and get all the latest information on new systems, services, applications, and whatever else is newsworthy. Or other things, such as what we hope to be the most complete Software Data Base for Cromemco systems available anywhere. We are building that now, and it is growing slowly. Please be patient.

What have we learned in the first three months of our existence? A great deal. For example, our members have let us know that software packages — especially packages for accounting, inventory control, process engineering, computer graphics, and anything to do with medical research and testing — are in great demand. We will deliver information on these areas in future issues.

We have also learned that our members are the types of members we hoped we would have. More than one hundred have indicated their willingness to supply I/O News with articles on subjects ranging from The [illegible] to [illegible] with an incredible array of topics in between.

Two members contributed very meaningful articles for this, our first issue. Dr. Lois Flynne, whose delightful feature on LISP (see front-page lead), and Darwin Engwer, with an extremely helpful CDOS applications note, have helped set the tone for the level of quality we are pledged to maintain. Their efforts are complemented by the announcements and technical articles meticulously prepared by Cromemco staffers Dr. David McLennan and D. Thompson McCalmont.

And while we're saying thanks, there are several people at Cromemco who have been avid fans of the IACU since they first heard the idea. In fact, there are too many to list here, but there is one, in particular, who has given generously of his time and resources to ensure that we got off to a good start. Thank you, Dr. Harry Garland. We will do our best to live up to the faith you have placed in us.

That promise extends to every member — everyone who has paid for a membership on faith. But, there is something each of you can do to help. Participate. Remember, it is only through the active, continued participation of its members that the IACU will reach its full potential as the vital and vibrant communications center we have all wanted for a long time.

Let's do it together.

Richard Kaye
Editor & Publisher

Cromemco's New Controller Card

Continued from first page

read or write single-sided, double-sided, single-density, or double-density diskettes. There are also commands to substitute memory with a string of bytes (ASCII, HEX or Decimal); or display memory contents in HEX with the corresponding ASCII characters displayed to the right of the hexadecimal listing; or search memory for a string of bytes (ASCII, HEX, or Decimal).

RDOS-2 also includes a printer driver so that output can be sent to the printer by means of the CTRL-P command

RDOS-2 output can be suspended at any time by the CTRL-S command. And input to RDOS-2 is now line-buffered so that you can delete the previous character with the backspace or DEL key — you can delete the current input line with the CTRL U command

SELF-TEST DIAGNOSTICS

A particularly exciting feature of the new 16FDC controller is in the system diagnostic software that comes as part of the RDOS-2 program. This lets you run a quick self-test of your system before booting up to assure that your memory, the 16FDC controller, and your disk drives are all working properly. Or if you are having difficulties with your system, the diagnostics can help you pin point precisely where the difficulty lies. The self-test diagnostic mode is entered simply by typing the letter "T" in response to the RDOS-2 prompt.

UP-GRADING YOUR PRESENT SYSTEM

By now you probably want to know how you can upgrade your system to quad-capacity with the new 16FDC. Here's how

If you have a single-sided, single density system, Cromemco recommends that you first up-grade to double-sided disk drives to increase your storage capacity Replacement double sided drives for your Z-2D or System Two carry the Cromemco Model number FDD T and can be ordered through your Cromemco dealer

Replacement dual 8-inch drives for the System Three carry the Model number 002D-50 or 002D-60 depending on whether your power line frequency is 50 hertz or 60 hertz

Once you have up-graded to double-sided drives, or if your system already has double sided drives, you can more than double your storage capacity once again by replacing your 4FDC with a 16FDC controller. The table shows the total storage available on 5-inch and 8 inch diskettes in single or double sided, single or double density operation.

AVAILABLE NOW — MAYBE

Since there will undoubtedly be heavy interest in the new 16FDC controller, your local dealer may be completely sold out of his initial shipment. If so, be sure to ask him to hurry off an order for you.

Cromemco expects to be able to fill all orders for the 16FDC within 30 days.

CROMEMCO DISK STORAGE CAPACITIES

	5-inch disk		8-inch disk	
	Single sided	Double sided	Single sided	Double sided
Single density	83K	173K	243K	594K
Double density	190K	390K	600K	121K

Breakthrough for the 80's

LISP Comes of Age

Continued from first page

For example, there is one LISP program that works at the level of a Ph.D in chemistry doing mass spectrographic analyses. Another serves as a physician's diagnostic assistant. Still others function as "assistants" in a variety of scientific, professional, educational and business applications, including programming itself. And, of course, LISP is the language behind the most advanced work done to date in computer understanding of natural language. For those who would like an account of some of these famed achievements, Pamela McCorduck's "Machines Who Think" provides an entertaining and highly readable history of the Artificial Intelligence field where LISP, after twenty-one years, still reigns as the prima lingua.

For most of us, unfortunately, LISP has remained hidden away in the remotenesses of think tank labs, which has acquired for it a reputation as arcane, a sleeping beauty of a language said to be guarded by thorny thickets of parentheses and memory devouring monsters with ravening ravines of recursions waiting to drag the unwary into bottomless pits. In other words, over the years, LISP has acquired a reputation as a bitch…elegant, yes, beautiful, yes, accomplished, yes…but so unattainable, esoteric and downright difficult that only someone with a touch of Marco Polo crossed with La Mancha and a strong masochistic streak would venture to quest it out. Of course, all of that is sheer stuff and nonsense. While it is true that LISP is beautiful, elegant and accomplished, and an exceptionally powerful programming tool, it is also easy to learn, practical and easy to use and now it is readily attainable — at least for we lucky Cromemco users.

In this, the year of LISP's majority, Cromemco leads the way into the 80's with the release of a powerful, extensive and extensible version of LISP that will dispel the silly fairy tales that have grown up around the language and let us get on with the practical business of putting LISP to work for us in a variety of applications.

Experienced LISP enthusiasts among us will not be disappointed. This is not one of those pale imitation micro LISPs you may have encountered before. This is a robust 30k LISP that provides you with 162 precision programming tools for fast and elegant list processing, plus string and character manipulating plus number crunching plus sophisticated file handling. And it runs at about a third the speed of a PDP10 which is zipping right along for an 8 bit machine — under 2 seconds to create and shuffle a deck of cards and deal and display five poker hands…as little as 132 seconds to compute and print the first 1000 primes…instantaneous location of a desired reference in a long list of references. (I ran these often used tests mainly to show that Cromemco LISP is by no means a slow coach.)

Moreover, it lets you conserve precious ram by leaving program modules out on the disk until needed and then frees up the ram used by the module. In other words, it lets you run programs too large to fit in one gulp into the tight 20k or so of user ram available in a 64k system running CDOS 2.35. You can create machine code subroutines you can access from LISP and you can save the LISP image, reincarnate, as it were, a LISP creation that can "remember" what it has "learned" from session to session.

In that 162 precision tool tool-kit, that together with other special symbols used gives you an extensive and extensible OBLIST of 245 symbols, you have many of the most advanced features of the language in its current state of the art form…powerful control functions, including IF, DO, CATCH THROW, SELF and SELECTQ as well as the standard COND, AND, OR and NOT for elegant and efficient recursion and iteration. The list, symbol, string and character selectors, constructors, recognisers and modifiers, including environment modifiers are as rich a collection as you could wish to find. All the standards you might expect are included plus…and, as you would also expect, a full complement of property list handlers. The function defining functions for adding new tools to the Cromemco LISP toolkit include not only the usual call by-value EXPRS and the special form FEXPRS but MACROS, which are a very powerful technique indeed, letting you pass programming details off to the machine. Powerful parameter passing mechanisms, allowing optional and auxiliary parameters to be specified in addition to the usual required parameters, make for clean and concise control. There are functions to perform and control evaluation, EVAL, PROG1, PROGN, QUOTE, etc. … functions to manipulate functions, APPLY, MAP, MAPLIST, even a simplified form of the LISP funarg. The rich collection of Input/Output tools, display tools and file handling tools make for streamlined communication between internal and external "sinks" and "sources" in sophisticated style with the full capabilities of the 3102 terminal at your command via CRT. Error handling tools readily permit quite complex debugging. Finally, a full array of arithmetic tools to handle fixed and floating point number crunching with the usual slew of logical functions is included. There's more, much more, but I fear to lose the "new-to-LISP" reader with all this LISP shop talk, though hopefully something of the richness of the Cromemco version of the language has come across. Suffice it to say that Cromemco LISP includes in the bloodlines of its pedigree such famous strains as MIT LISP, Stanford LISP, InterLISP and V LISP to drop a few names. It is truly a LISP to grow with into the eighties.

While large scale LISP programs on the order of those you read about in the Artificial Intelligence literature will have to await things like Cromix with increased user memory capacity and the next wave of other Cromemco developments waiting down the track a piece, some very decent mid-range LISP creatures can be produced right now on your Cromemco as it stands and the building blocks for the larger ventures can be developed ready to go. Moreover, Cromemco LISP is a great teaching LISP — whether you want to teach others, or, teach yourself.

In all ways, there's more than enough practical stuff

you can do with Cromemco LISP right now. As Alan O Neill put it to me when I was futzing around, hesitating on whether to wait for what might be coming out tomorrow, or, tomorrow, or, tomorrow — "Tomorrow," said Alan, "never gets here. Tomorrow is always the next day. Meanwhile, you miss out on all you could be doing today." I've been having fun for many todays working with today s Cromemco LISP on my today s Cromemco System III Thank you Alan. Incidentally, Alan is Cromemco's Customer Support Manager and member Numero duo in the International Association of Cromemco Users. I heard the godfather beat him out for the number one slot but he said it was Harry Garland

Thanks are also due to Roger Melen and the team at Cromemco who have made such an excellent LISP as this available to us for our pleasure and profit — and special thanks to the genius who created this Cromemco version Of course smart products like this are what we've come to expect of Cromemco.

For the LISP enthusiast...an open sesame to a veritable treasure trove of LISP tools...the sort you expect in the best LISPs...

Let's turn now to the neglected LISP novice to whom the rest of this article is directed. For the novice, Cromemco LISP is a mild mannered, easy to code, marvelously buoyant language You can jump right in and float. In some ways, learning Cromemco LISP is a little like playing Adventure ... lots of fun and addictive. Let me try to communicate a little of what it is like to use Cromemco LISP taking a sort of Berlitz approach. Of course, what you will ultimately do with your new-found language skills is a function of your interests and imagination aided by what has already been done in Artificial Intelligence work. Cromemco LISP is simply as already said, a kit of precision programming tools you can use to build programs and more tools to build programs for a variety of purposes — 'smart' word processors, content analysers, manager's assistants, law clerks, reference librarians, debuggers, symbolic mathematisers, teacher's helpers, game players. More than enough rewards await you to make learning LISP well worth your time and effort. Now on with the promised Berlitz blitz overview of Cromemco LISP.

Both LISP data structures and programs are written in the form of symbolic expressions. Practically anything you can think to type in at the terminal has the potential to be a LISP symbolic expression, given the proper punctuation with appropriate use of spaces and parentheses and special characters like the back slash, \, the single quote, ', the period, ., the semi-colon, ; and a pair of double quotation marks, "" all of which play a special part in the Cromemco LISP syntax. The semi colon is a comment character, like REM in Basic. It lets you insert comments into a program. The use of the other four we will explain below. Cromemco LISP recognizes five types of symbolic expressions:

1. Numbers — same as in any other language. Cromemco LISP can handle up to 14 bit signed integers and up to 32 bit signed floating point numbers.

2 Literal atoms — any collection of one or more letters, digits, or characters as long as the first one is a letter and there aren't any separators like spaces, commas, or, periods in between. Every word in this sentence, from word 1 to word-n, is a literal atom.

3 Strings — any collection of zero or more letters, digits, or characters enclosed in double quotation marks. "Anatomy of LISP" is a string and coincidentally a good book on LISP that will help you understand the abstract principles of the language. "Artificial Intelligence Programming" is also a string and a book that will have you writing sophisticated LISP programs in no time. It is delivered as part of the package with the Cromemco LISP diskette which, of course, also comes with a manual. As final examples, "2 * two plus 2.2" is also a string, so is "!!!!!???".

4. Characters — any single letter, or digit, or, character preceded by a back slash. \s or \? or \5 or \(or \). See, those dreaded parentheses are just harmless characters. They are used to create the fifth type of symbolic expression which is...

5. Lists — which consist of a left parenthesis, followed by zero or more of any mix of the five types of symbolic expressions, followed by a right parenthesis. Lists are the sentences of LISP! Just as sentences can have clauses, so can a list be ((a list) of (lists)). Just remember every left parenthesis (((must have) a matching) right parenthesis). As soon as Cromemco LISP encounters as many closing right parentheses as leading left parentheses, it jumps in and starts trying to make sense of the "sentence" you typed in. It is sort of nice not to have to count out the trailers.

Since lists are the essence of LISP, which is actually an acronym for LISt Processing, let's go into a little more detail. First, we need to explain dot pairs and the use of the period. If a period occurs in the middle of a set of digits, LISP reads it as a decimal point... 2 35 or, 99 9876 for example. Otherwise a period means a dot-pair. All lists reduce to dot-pairs. For example, the author of "Anatomy of LISP" could be represented as (John Allen), which in dot-pair notation is (John.(Allen.NIL)). (John.Allen) is not the same data structure as (John Allen), which is structured as just shown. Mostly you will express programs and data in list notation, so for now don't worry too much about dot-pairs. For now, just realize that NIL lurks at the end of a list list as an end of list marker as it were. As for those parentheses and their placement, think of

parentheses as a syntactical convention that give structure to LISP "sentences", much the way punctuation gives structure to sentences in a natural language.

Now, numbers, strings and characters are constants, that is Cromemco LISP takes such symbolic expressions literally. "East" is "East" and \e is \e and 4 is 4, though four could be 4 or (four thousand), or, "east" or, *, or, a program module that quadruples things. Literal atoms are variables. If you want a literal atom to be taken literally you have to quote it a la 'four or (quote four)...'four is shorthand for (quote four). Same goes for lists. If you want a list to be taken literally, quote it one way or the other. Otherwise LISP will try to read meaning into literal atoms and lists. Those 162 programming tools Cromemco LISP comes with are literal atoms whose meanings are the attached program code. For example, ADD is add in the sense of add up a set of numbers. (ADD 1 2 3 4 5) is 15 though 'ADD is ADD and (ADD 1 2 3 4 5) is (ADD 1 2 3 4 5). (FIRST '(John Allen)) is John and (REST '(John Allen)) is (Allen). If you wanted to inform Cromemco LISP that John Allen is the author of "The Anatomy of LISP", then you might use another of the Cromemco LISP tools as follows

(PUTPROP 'John Allen 'Author of "Anatomy of Lisp) in which case LISP would put the information (Author-of, "Anatomy of LISP") on the property list of the literal atom John-Allen. Later, you could dig up the info on John Allen by asking LISP (GETPROP 'John-Allen 'Author of) and you would be told ..."Anatomy of LISP". We will be returning to PUTPROP and GETPROP, and the other Cromemco LISP property list functions later. Every literal atom has a property list where you can associate attributes and values with the literal atom as we just did.

Obviously, what we were doing could be a start on setting up a reference and retrieval system. The primary tools we need to create such a system come built into Cromemco LISP, which as a language is a "natural" for associating items of information and retrieving items of information, either by direct or indirect associations. After a few sessions with Cromemco LISP, you could easily write a little program to cross-reference a whole slew of bibliographic references and retrieve information by author, title, or, subject. Towards the end of this article, we'll take a crack at a start on this. It won't be wildly efficient, or, elegant, or anything approaching what Cromemco LISP is capable of with all stops out. However, it should give you a feeling of what it is like to program in LISP and show you that Cromemco LISP is a sweetheart and not the vicious virago of legend. If the sweetheart image isn't the one that comes across, blame me and go find someone who can communicate, or, better yet buy the Cromemco LISP diskette and teach yourself...it's that easy!

Here goes.

Imagine you are sitting at your terminal. The Cromemco LISP diskette is in the drive. Type LISP. The screen clears, the Cromemco LISP banner message is displayed and the prompt > appears in the upper left hand corner of the screen. From now until you exit LISP by typing (EXIT) you will be conversing with EVAL. EVAL is a program that evaluates symbolic expressions. EVAL takes one symbolic expression and returns another symbolic expression as the value of the first.

Let's try a few warm up exercises. Type in a number. An integer will do for starters. EVAL responds with the integer you typed in. Type in a floating point number. EVAL again responds with what you typed in. EVAL recognizes numbers as constants. Constants evaluate to themselves. A constant is a constant is a constant. EVAL has a touch of Gertrude Stein...so gaily forward. Type in Gertrude. EVAL replies unbound variable Gertrude. Ooops, Gertrude is a a literal atom literal atoms are variables. Try 'Gertrude, which as you know is shorthand for (QUOTE Gertrude). QUOTE, or, ' makes Gertrude a constant and not a variable. QUOTE is a LISP tool that tells EVAL to take the symbolic expression following the ' literally. In LISP talk, it suppresses evaluation. Moreover (QUOTE Gertrude) is a LISP symbolic expression and a program module! Every symbolic expression can be a LISP program as long as the semantics are correct. We could make Gertrude into a LISP function. Just for the fun of it, let's do just that.

```
(DE Gertrude (noun repeats)
   (IF (ZEROP repeats) (LIST 'a noun)
   ;that means if the value of the variable repeats is
zero,
   ;then return a list made up of a and the value of the,
   ;variable noun. Otherwise do what follows below ... ;
   (APPEND (LIST 'a noun 'is) (Gertrude noun (sub1
repeats)))
   ;which means make a list of a, the value of noun and
is
   ;and put that list onto the result of applying Ger-
trude;
   ;to noun and repeats minus 1; ))
```

EVAL evaluates that little goodie and replies GERTRUDE. You have just writen your first LISP function and a recursive one at that. Now LISP has a meaning for Gertrude. The meaning of Gertrude is the function definition you've attached to Gertrude, though 'Gertrude still means Gertrude. DE is the tool you use to make functions that require a fixed number of arguments, specified in the formal parameter list which in the case above is (noun repeats). Frankly, if you draw on what you know of function definition in a language like Basic, you'll get the general gist, at least in the beginning. Since computing the factorial of a number is a hoary old exercise we are all familiar with, here is a LISP factorial function definition, which you can compare to the recursive definition in Basic given in the 32k Structured Basic Manual, page 258.

```
  (DE Fact (n)
     (IF (ZEROP n) 1
     (MUL n (FACT (SUB1 n)))))
;MUL is the function that multiplies ... as in (MUL 2 3
4) is 24;
```

Now back to Gertrude ... let's apply Gertrude to some actual parameters. Give EVAL (Gertrude 'rose 3). EVAL should respond ... (A ROSE IS A ROSE IS A ROSE IS A ROSE) ... very Gertrude Stein. Boring! However, before we leave Gertrude, let us show you how to define Gertrude iteratively. While recursion in LISP is

beautifully elegant and succinct as you can see from the above, iteration is in some cases more efficient, particularly where you might be constrained for memory. Recursive Gertrude would overflow the stack at around 65 roses. So here is ITER GERT ... using the Cromemco LISP DO loop. The code should be pretty self-explanatory, though the placement of the parentheses may be a bit puzzling. Don't worry. The Cromemco LISP manual provides a formal description of each tool which shows you how to place the parentheses to be syntactically correct. After a little practice, it comes as naturally to you as everyday simple punctuation, or, the syntactical conventions of any other programming language. Far too much hoo-ha has been made around LISP's parentheses.

```
  (DE Iter-Gert (noun repeats)
     (DO ( (repeats repeats (SUB1 repeats))
;keep on doing this:(sentence (LIST 'a noun)
;until repeats is zero; (APPEND (LIST 'a noun
                                    'is)sentence)))
     (((ZEROP repeats) sentence))))
```

On with the warm up. Type in (ADD 1 2 3 4 5). EVAL replies 15. ADD is a function that takes an indefinite number of arguments. Showing you how to cre8te those sorts of functions is a bit too complex for starters. However, we can use what we learnt from Gertrude to get started on building the reference associator and retriever. Let's start with some exercises with the tools we've just used and the tools we will be using.

For the novice LISPer...an easy entree to a most powerful programming language for both systems and applications work...

Type (LIST 'Artificial 'Intelligence 'Programming). EVAL will respond (ARTIFICIAL INTELLIGENCE PROGRAMMING). Now try (APPEND (LIST 'Eugene 'Charniak) (Artificial Intelligence Programming)) and the result will be (EUGENE CHARNIAK ARTIFICIAL INTELLIGENCE PROGRAMMING). Next, (SETQ Author (Eugene Charniak)) and you will have given the variable Author the value (Eugene Charniak). In the same fashion attach the above title to the variable Title. Now use another constructor function ... CONS (SETQ Part ref (CONS Author Title)) and you will get the following...

((Eugene Charniak) Artificial Intelligence Programming) (FIRST Part ref) is (Eugene Charniak) and (REST part-ref) is (ARTIFICIAL INTELLIGENCE PROGRAMMING). (REST (FIRST Part ref)) is (Charniak). (REST Title) is (INTELLIGENCE PROGRAMMING) and (REST Title 2) is (PROGRAMMING). REST can take an integer as an optional second argument — a very nice little touch.

Let's try our hand with one more reference just for good measure, so we get the hang of FIRST and REST and the other few tools. (SETQ REF1 ((John Allen) (Anatomy of LISP)(McGraw Hill 1979))) EVAL prints out the reference. Now type (FIRST REF1) ... gives you (John Allen). Type (FIRST (FIRST REF1)) ... gets you John. Try (REST REF1) ... gets you ... ((Anatomy of LISP)(McGraw Hill 1979)). Try (REST (REST REF1)). Aha ((McGraw Hill 1979)). Try (REST(REST(REST REF1))) NIL says EVAL. (NULL (REST(REST (REST REF1)))) T says EVAL. T means true in LISP talk.

As just mentioned, REST can take an integer as a second optional argument — very useful indeed. (REST REF1 2) will get us ((McGraw Hill 1979)). (REST REF1 3) will get us NIL.

Now for CONS one more time. Let's do this first (SETQ REF ORDER '(Author Title Publisher)). By now you know what EVAL will respond. Now try (CONS (FIRST REF-ORDER)(First REF1)). This time you get (Author John Allen) ... and hopefully you are getting the hang of LISP.

On with our dog-paddling. Cromemco LISP really is very buoyant. Flow along with it and in no time your dog-paddling will turn into the Australian crawl. As our final warm-up before getting down to serious business, let us define one more example function to put REF1 and REF-order together.

```
  (DE ref-maker (refrence format)
     (IF (NULL refrence) NIL
     (CONS (CONS (FIRST format) (FIRST reference))
     (ref maker (REST refrence)(REST format))))))
(REF-MAKER REF1 REF-ORDER) should give us...
  ((AUTHOR JOHN ALLEN) (TITLE ANATOMY OF LISP)
     (PUBLISHER McGRAW HILL 1979))
```

If for some strange reason, we happened to like any of these monstrosities we've created so far — namely, GERTRUDE, ITER GERT, REF-MAKER and even REF1 and REF-ORDER — Cromemco LISP makes it simple! On your Cromemco LISP diskette, you will notice a file named AUTO.LSP. It contains functions for making Auto-load files. Actually, what you have in this little package is a way to conserve precious memory by "virtualising" large programs. You can read the full details in the Cromemco LISP manual. All we are going to do here is show you how to load a file and write functions out to a file. You can streamline the procedure down the track. Tailoring everything to suit yourself is the very essence of LISP programming. But we start with what's there. Type (LOAD "AUTO.LSP"). EVAL reads and evaluates the file and lists the names of the things defined on the file. MKAUTO is the function we want. We now type (MKAUTO "Myfile" '((Gertrude keep) (Assigner.keep) (Ref-maker.keep) (Ref1.keep)(Ref order.keep))) and our file gets written. To load it in later type (LOAD "MYFILE.ATO") and that's it

This article is starting to get awfully long and we are still puddling around in the shallows. Problem is LISP is an ocean. Let's take the plunge into some deeper waters and write that promised cross-reference and retrieval program for a list of bibliographic references. Ultimately, it would be very nice to just have to type in for example, (CROSS REF BIBLIO :by: FORMAT) and (RETRIEVE :stuff by: 'JOHN ALLEN) and (RETRIEVE :references on: 'ARTIFICIAL-INTELLIGENCE) and (TRACKDOWN 'EUGENE CHARNIAK 'PUBLISHER) and carry on other pigdin type conversations with our

Cromemco LISP system. Can be done. While it may be a little deep water for an introduction, let's take a deep breath, jump in and pray.

To simplify things, let's assume each bibliographic reference is in the form of a list of literal atoms as follows:

(author title publisher date subject)

Example would be (JOHN-ALLEN ANATOMY OF LISP MCGRAW-HILL D:1978 LISP). (AUTHOR TITLE PUBLISHER DATE SUBJECT) is the format of the reference. A bibliography would be a list of these references a la

((Eugene Charniak Artificial-Intelligence-Programming Lawrence-Erlbaum D:1980 LISP)

(Pamela-McCorduck Machines-Who-Think Freeman D:1979 LISP)

(John-Allen Anatomy-of-LISP McGraw Hill D:1978 LISP)

;etc.etc.;)

Let's further assume we only want to be able to retrieve a reference by author, or, title, or, subject. So what we want is to associate a title with both its author and subject and associate the author, publisher, date, and subject with the title. Our starter system here won t be able to handle multiple authors, or, multiple subjects for the one title. However, it will be able to handle multiple titles for any one author or subject. The limitation mentioned is purely for simplicity's sake and is no reflection on the capabilities of Cromemco LISP.

So, herewith the program, heavily commented so you can follow what it does.

****SPILL — A Simple-minded Personal Information Library Looker-upper**

First, let's deal with the function that cross-references the bibliography, which will be in the form just specified, according to the format just specified, with each reference inter-related as described. This function CROSSREF applies two other user-defined functions, DESCRIBE-REF-ELEM and CROSS-ASSOC, to each reference in the bibliography successively until the end of the bibliography is reached.

```
(DE CROSSREF (BIBLIO FORM)
  (DO ((BIBLIO BIBLIO (REST BIBLIO)))
  (((NULL BIBLIO) (QUOTE FINITO)))
  (DESCRIBE REF ELEM (FIRST BIBLIO) FORM)
  (CROSS ASSOC (FIRST BIBLIO) FORM)))
```

DESCRIBE-REF-ELEM is a simple recursive function that handles the business of designating what type each of the elements of a reference is ... John-Allen is an author, McGraw-Hill is a publisher, etc. We use this property list information later in our retrieval function.

```
(DE DESCRIBE REF-ELEM (REF FORM)
  (IF (NULL REF) NIL
  (PUTPROP (FIRST REF) (QUOTE TYPE OF) (FIRST
FORM))
  (DESCRIBE-REF-ELEM (REST REF) (REST FORM))))
```

CROSS ASSOC handles the busy work of adding the actual title of the given reference onto the property list of the given author and the property list of the given subject. It also puts the actual author, publisher and date under the appropriate attributes onto the property list of the actual title. We use the Cromemco LISP tool ADDPROP, instead of PUTPROP in the case of the author and subject because of the multiple titles possibility mentioned above. The Cromemco LISP tool PROGN is used to let us carry out a number of operations where only one would normally be allowed. We use SET and not SETQ, which we showed you earlier, because in this case we want the first argument of the function to be evaluated and SETQ quotes its first argument. SETQ is simply (SET (QUOTE argument-1) argument-2)

```
(DE CROSS-ASSOC (REF FORM)
  (DO ((REF REF (REST REF))
  (FORM FORM (REST FORM)))
  (((NULL REF)
  (PROGN (ADDPROP AUTHOR (QUOTE TITLE) TITLE)
  (PUTPROP TITLE (QUOTE DATE) DATE)
  (PUTPROP TITLE (QUOTE PUBLISHER) PUBLISHER)
  (PUTPROP TITLE (QUOTE AUTHOR) AUTHOR)
  (ADDPROP SUBJECT (QUOTE TITLE) TITLE))))
  (SET (FIRST FORM) (FIRST REF))))
```

And that about does it for the cross-referencing part of our program. We simply call CROSS-REF with the actual list of references and the actual format and we are in business. Cromemco LISP would now "know" that, for example, John Allen was an author. It would "know" the titles of works by John-Allen. It would "know" the author, publisher and date of any given title in the bibliography and it would "know" what titles went with what subject. As pointed out earlier, this is a simple-minded program, restricted to a narrowly specific domain and not as elegant and general as it might be. However, this is a first lesson in Cromemco LISP. Aim was to do something that would be reasonably easy for a beginner to follow ... and it seems to me that fairly concrete examples are easier to follow at first. That said, let s move on to the retrieval end of the game.

For the educator...a clear and cleanly coded teaching LISP with advanced state-of-the-art features that let you demonstrate LISP's pyro-technics.

RETRIEVE is the function that retrieves bibliographic information. It takes one argument, a single literal atom, which could be the name of an author, or the title of a reference, or a given subject. For example, we could (RETRIEVE ;works by; Pamela-McCorduck), or, (RETRIEVE ;references on; Artificial Intelligence), or, (RETRIEVE ;the full listing on; MACHINES-WHO-THINK). RETRIEVE uses a few new Cromemco LISP tools. To deal with these in the order they occur in the function:

&AUX let's you specify auxiliary variables in the formal parameter list. In this case the auxiliary variable receives the value of author, title, or subject depending on the type of the argument given to RETRIEVE.

PLIST returns as its value the full property list of its argument. If the actual value of THING is a title, then

RETRIEVE simply returns the full property list of the actual title — author, publisher and date.

LET is a little like LET in Basic. For now, just think of it in those terms. If the THING is not a title, which means it has to be an author, or, a subject, then TEMP is assigned the value of all titles on the property list of THING. If there is only one title, and ATOM is the function that checks for this since a single title is a literal atom then the value of RETRIEVE is the property list of the title found. Otherwise, we enter the DO-loop and retrieve the property lists of all titles found.

REVERSE reverses a list and is used here to get the information out in order. Only GETALL remains to be explained and that is done below since GETALL is a user defined function.

Hopefully, you will now be able to follow RETRIEVE as shown.

```
(DE RETRIEVE (THING &AUX (TYP (GETPROP THING
(QUOTE TYPE-OF))))
    (IF (EQ TYP (QUOTE TITLE)) (PLIST THING)
    (LET ((TEMP (GETALL THING (QUOTE TITLE))))
    (IF (ATOM TEMP) (PLIST TEMP)
    (SETQ INFO NIL)
    (DO ((TEMP TEMP (REST TEMP)))
    (((NULL TEMP) INFO))
    (SETQ INFO (CONS (LIST(FIRST TEMP)
    (REVERSE (REST(REVERSE (PLIST (FIRST TEMP)))))
    INFO))))))
```

GETALL is fairly straightforward. It is essentially just like GETPROP, only it collects all values associated with a given attribute on the property list of an object. If there aren't any then the value of GETALL is the original value of VALIS which starts out as NIL; if the attribute assigned to TEMP is EQual to the sought ATTR then CONS puts the corresponding value together with the old value of VALIS and we iterate around the DO-loop until we reach the end of PROPLIS and the value of VALIS is returned as the result of GETALL.

```
(DE GETALL (OBJ ATTR & AUX (PROPLIS (PLIST OBJ))
(VALIS NIL))
    (DO ((PROPLIS PROPLIS (REST PROPLIS)))
    (((NULL PROPLIS) VALIS))
    (SETQ TEMP (FIRST PROPLIS))
  (IF (EQ (FIRST TEMP) ATTR) (SETQ VALIS (CONS (REST
TEMP) VALIS))
    NIL))))
```

And that essentially winds up SPILL.

As a final farewell to this Berlitz-blitz, consider a situation where you want to track down certain information on a given item where that information is not directly associated with the item though it may be indirectly associated. For instance, you may want to correspond with the author of a text ... TRACK-DOWN John Allen, for example, who is the author of Anatomy-of-LISP. There is no address to be found on the property list of the given author. However, if you happened to have fed Cromemco LISP a publisher's directory associating an address with each publisher, then the function TRACK DOWN given below would retrieve the addresses of the publishers of works by the author, which would at least be a starting point. TRACK-DOWN works a little the way we might when we contact a friend to contact a friend of the friend to contact an acquaintance of the friend of the friend to get the telephone number of someone we need to get in touch with.

First of all TRACK-DOWN looks for the requested information on the property list of the object we want the information about. If that is successful, then it returns that FIRSTRY as the result. Otherwise it uses the function VALIST to collect all values on the property list of the object and iterates and recurses through each of those values and the values found on the property list of each value until it finds the information, or, runs out of all possible leads. While not the greatest example of what Cromemco LISP can really do, try doing TRACK DOWN in Basic, or, Pascal, or, whatever other high level language you prefer in ten lines of code! Make that fifteen with the addition of GETALL. I think it was Charniak who said LISP was a lazy programmer's language. Why not?

```
(DE TRACK-DOWN (OBJ ATTR &AUX (FIRSTRY
(GETALL OBJ ATTR)))
    (IF FIRSTRY (CONS OBJ FIRSTRY)
    (LET ((OBJVALS (VALIS OBJ)))
    (DO ((OBJVALS OBJVALS (REST OBJVALS))
    (NEXTRY NIL (CONS(TRACK-DOWN(FIRST OBJVALS)
ATTR) NEXTRY)))
    (((NULL OBJVALS) NEXTRY))))))

(DE VALIS (OBJ & AUX (LIS (PLIST OBJ)) (VALS NIL))
    (DO ((LIS LIS (REST LIS))
    (VALS VALS (CONS (REST (FIRST LIS)) VALS)))
    (((NULL LIS) VALS))))
```

So ends the Berlitz-blitz. Hopefully, it has given you some glimmering of Cromemco LISP ... and do remember, it is only a glimmering. We have only skimmed 30 or so of the 162 tools available, flown lightly over a little of the syntax and semantics and written a few "Dick and Jane" functions. I strongly suspect that the best way to learn LISP is to spend some pleasant hours conversing with a good version of the language ... and, as I've been saying throughout Cromemco LISP is a beauty. Enjoy."

Aerial view of the author

Dr. Flynne is an Australian who started out to be a classical Greek and Latin scholar; got diverted; found advertising more fun and lucrative; did that for nine years; picked up various television advertising awards got bored; came to the United States to pursue a doctorate in Mass Communications; had a brief encounter of the worst kind with Fortran; discovered IPLV; discovered LISP; fell madly in love; did Basic, re-did Fortran, did Pascal; didn't care; last year discovered Cromemco; discovered Cromemco LISP; fell more madly in love. Currently in hock to the bank. Last seen hooking a modem to her system III and studying Electronic Fund Transfer...needs another 64kz and possibly a lawyer, as current captive one refuses all responsibility.

In serious moments Dr. Flynne teaches LISP, (all too rarely), Basic, (all too frequently) and various and sundry research methods and social issues courses at San Francisco State University.

Configuring the Summagraphics Bit Pad One To Conform with Cromemco SDI Software

Cromemco's new SDI interface includes a software package which permits interaction between a digitizer tablet input and the SDI's color graphics display. The software assumes that the digitizer tablet is a Summagraphics Bit Pad One with an RS 232 interface connected to I/O port 20h. The equipment required to utilize the digitizer pad with the SDI is:

1. Cromemco TU-ART
2. Three wire RS 232 cable
3. Summagraphics Bit Pad One, RS 232 mode
4. Power supply

Note that a separate power supply is required for the digitizer tablet and is available from Summagraphics.

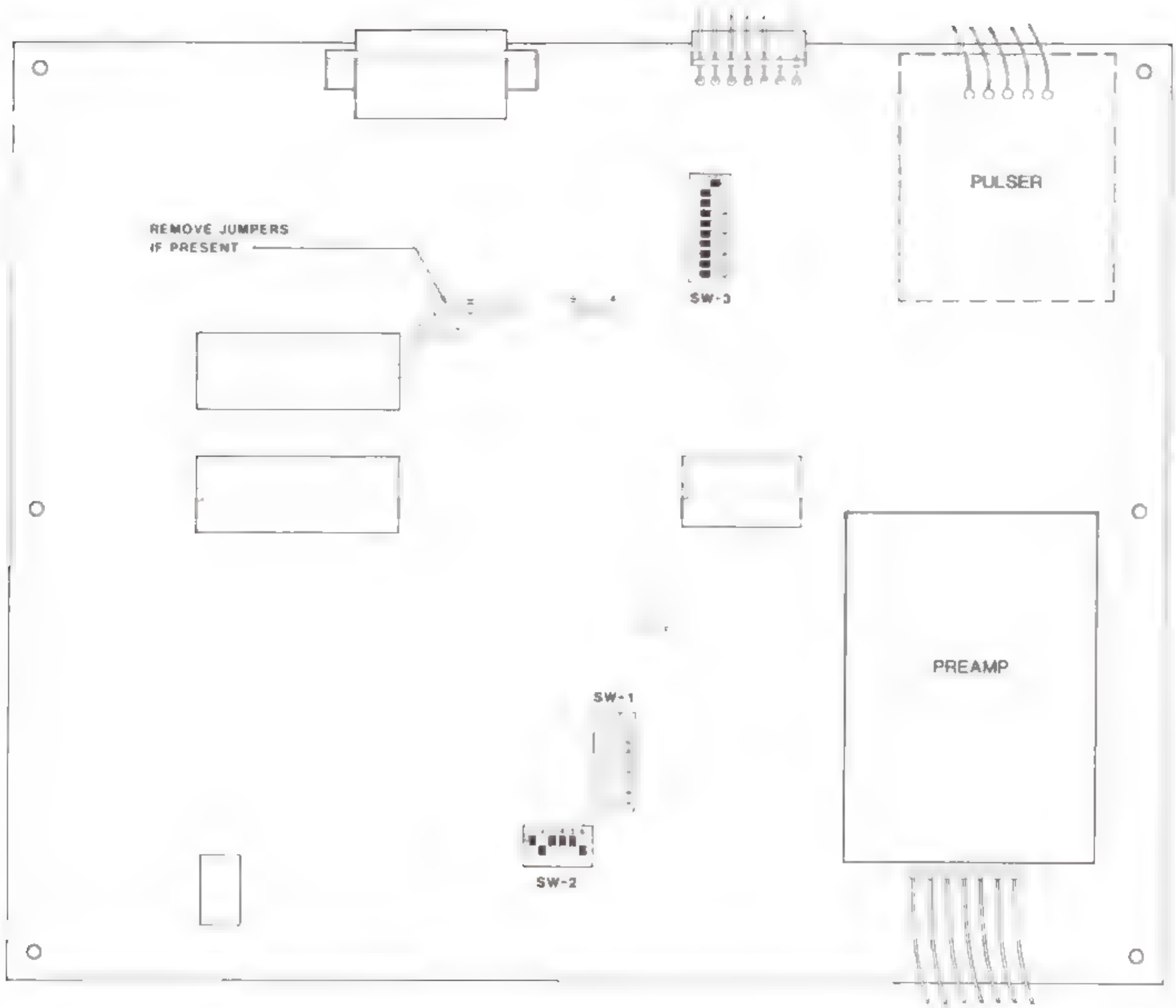
The digitizer tablet, as supplied by Summagraphics, is set for a baud rate of 9600, odd parity, and one stop bit. These settings should be changed to:

1. 19,200 baud
2. Even parity
3. Two stop bits

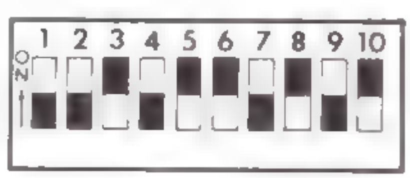
To change these settings the bottom of the digitizer tablet must be removed to expose the printed circuit board. The baud rate is controlled by switch settings while the parity and stop bit are selected by the presence or absence of jumper wires on the board. Referring to the following diagram, set switches SW 2 and SW 3 as shown and remove jumper wires as indicated. Switch SW 1, used for calibration, is factory set. Its settings should not be changed.

The cable from connector of the TU-ART board to the digitizer tablet should contain three (3) wires only and be connected as shown in Figure 8 of the Cromemco TU-ART Digital Interface Instruction Manual (part no. 023-[illegible]). The appropriate switch settings are:

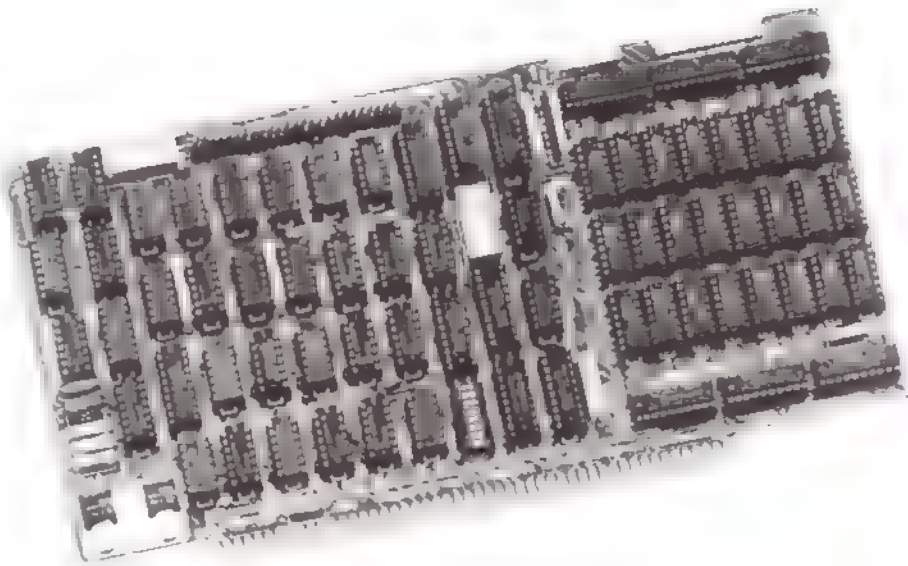
This will address device A at port 20h and device B at port 50h. In setting the address of device B the only [illegible] is that it not conflict with the address of other I/O devices.



Bit Pad One PCB configuration



16KTP and 48KTP Two Port Memory Boards

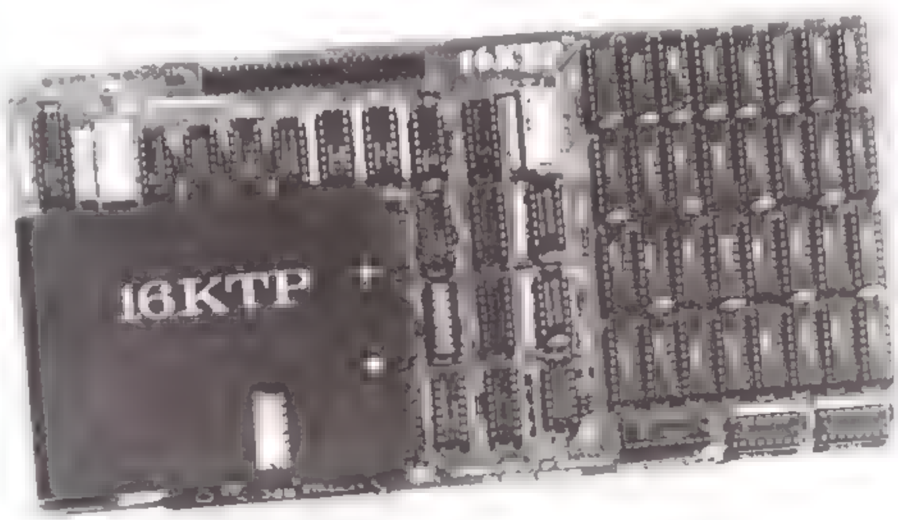


Cromemco introduces two new two-port memory boards, the 16KTP and the 48KTP, for use with their Model SDI Color Graphics Interface. The SDI is a high resolution, color graphics interface which can be used to display images with up to 754 x 482 point resolution. This interface and the two-port memory, when used in conjunction with an RGB color monitor, turn any Cromemco computer into a highly sophisticated graphics system with features unparalleled in the industry. This graphics system is ideal for a wide range of applications including process control, medical imaging, computer-aided instruction, and business statistics and charts

The two-port memory gives independent high-speed access to the computer memory. These two-port memory boards have two sets of address and data lines which give them the ability to process the SDI's memory refresh requests while the CPU simultaneously and independently executes a user program. Picture information is accessible by the SDI through a connector on the top of the memory boards. This direct connection of the SDI and the two-port memory bypasses the S-100 bus so the CPU accesses the two-port memory as though the SDI were not present. Consequently, use of the special two port memory in a graphics system assures 75% to 100% CPU utilization, depending on the application software

These two-port memory boards are also designed to work with Cromemco's powerful graphics software package. This graphics package provides a full range of powerful, human oriented commands that operate from such common high-level languages as BASIC, FORTRAN and RATFOR The graphics software package will operate with one or two pages of two-port memory. Two pages of 48K bytes of RAM are required for complete utilization of all available software options.

Cromemco's two-board color graphics interface (Model SDI) is available for $595. The 16K two-port memory board (Model 16KTP) is available for $795 and the 48K two-port memory board (Model 48KTP) is available for $1785. Cromemco's graphics software package is available on either 8" (Model SGS-L) or 5" (Model SGS-S) floppy diskette for $295. For additional information, contact Cromemco, Inc., 280 Bernardo Avenue, Mountain View, CA (415) 964-7400



The Cromix Operating System

Continued from first page

drives; allocating the computer's memory to various processes and/or users; and partitioning CPU time, devoting intervals, or **slices**, of that time to the processes currently being executed.

An operating system may perform these functions to varying degrees, depending on its complexity. Many operating systems presently being used on microcomputers are **single-user**, which means that CPU time is devoted entirely to the processes of that one user. Other operating systems are **multi-user**, allowing more than one user to use the computer simultaneously, and/or **multi-tasking**, allowing more than one process to execute at one time.

Many readers are already familiar with Cromemco's first disk operating system, CDOS, a single-user operating system which has become a hallmark of the industry for ease of use and versatility. Cromemco will soon begin shipments of its new multi-user, multi tasking **Cromix operating system**, which offers even greater flexibility, an outstanding file management system, and complete support of all Cromemco hard disk and floppy disk products. In addition, the Cromix operating system has one of the fastest speeds of operation of any operating system for any microcomputer.

The Cromix operating system has been a two year project developed entirely by engineers at Cromemco. It is similar in many ways to the well known Bell Labs operating system, Unix†. This article will present some of the outstanding features of the Cromix operating system

Cromix Memory Allocation

The address space of the 8 bit microprocessors used in most present-day microcomputers is limited to an addressable range of 64K (65,536) 8 bit words (or bytes) of RAM Since the CPU is not designed to address more than this address space at a given instant in time, various methods of **memory management** have been devised to allow the computer to execute multiple processes.

One excellent method for doing this is called **bank switching**. For this method the RAM of the computer is divided into separate pages (or banks) of 64K each, only one of which is turned on at a given moment in time. The CPU then has the capability of changing banks (i.e., deselecting one bank of RAM while simultaneously selecting another) to change the process currently under execution. This procedure is done many times each second, thus allowing the CPU to give the appearance of executing multiple processes simultaneously

Cromix uses bank switching for memory allocation, devoting one bank of RAM (or 64K) to the Cromix operating system itself and one bank of RAM to each concurrently executing process. A **process** is defined as any user executing any task; it will always be allocated its own bank of memory. Thus, the following is an idealized picture of the main memory of a microcomputer which is running the Cromix operating system:

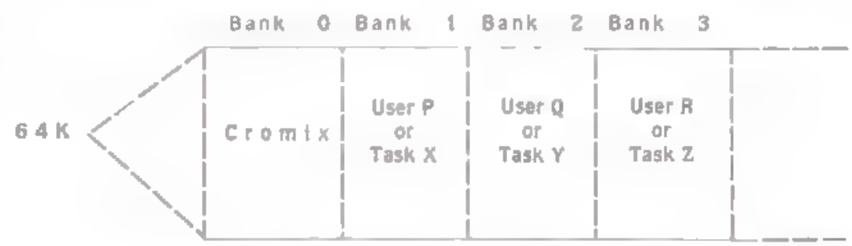


Fig. 1. Cromix Memory Allocation.

A computer system containing hardware for a fully implemented Cromix system contains 448K of RAM and allows up to 6 users or tasks in any combination. Bank 7 of such an implementation is reserved only for system initialization and no actual RAM is assigned here once the Cromix operating system is running

Cromix allocates memory for users and tasks as it is required. Users who are not currently executing any programs are said to be in the Cromix **Shell**, which means they are under the control of Bank 0 in Fig. 1. As soon as such a user executes a program or task, Cromix assigns another bank of memory to that process.

Suppose, for example, that users 1 and 2 are currently logged on and that user 2 is not executing any programs. If user 1 begins two processes, these will execute in Banks 1 and 2 of RAM If user 2 now begins a process, it will execute in Bank 3.

It is apparent that this is an excellent method of memory allocation as it allows the most efficient use of both memory and CPU time. Memory is assigned only when needed and then, in the order in which requests for it have been made. The mechanism used to execute multiple processes is described further in a later section

File System

The file management system of the Cromix operating system is a versatile tree structure of files and directories. A file down any branch of this tree may be accessed by means of its path name. This file access scheme also has one of the fastest speeds of operation obtainable.

Tree Data Structure

Data is organized by an operating system into **files**. A file may be loosely defined as a sequence of bytes stored in the mass memory of the computer. Cromix manages files through a powerful system known as **hierarchical directories**. This means that all files are contained in and organized by a tree structure of directories, sub-directories, and files. A **directory** is a collection of files and is itself a type of file as

†Unix is a trademark of Bell Laboratories

we shall see later.

The directory at the most basic level is called the **root directory**, meaning the one from which all other directories and files must grow and be accessed. Sub-directories or files may "branch out" from this root directory. Sub-directories of one level may contain sub directories or files for the next level down in the tree. An example of this structure is shown in Fig. 2.

Fig. 2. Cromix File Structure.

The tree structure described in Fig. 2 above is known as the Cromix **file system.** The root directory is known as the **ancestor** or **parent** of all sub-directories and files. Any sub directory or file is the **descendant** or **child** of the directory which is one level higher in the tree. This information will prove useful in describing path names in the next section

The importance of the Cromix tree file structure becomes most apparent in actual use. Its high speed of file access and flexibility in device I/O make the tree data structure an excellent file management scheme for a multi-user, multi tasking operating system.

In particular, note that all I/O devices are members of the file system. Thus, rather than accessing a disk or other mass storage device as a unit, it is referenced only indirectly by accessing a file which happens to be found on that unit. The significant quantity is the actual file desired; its location will be automatically determined by the operating system.

Path Names

Any type of file is accessed under the Cromix operating system through a **path name.** The term file here is used to describe one of four things: a **directory**, a **character device** (includes sequential I/O devices such as consoles and printers), a **block device** (includes block I/O devices such as floppy or hard disk drives), or a **data file** (includes executable, binary, and ASCII files). In general, the data files are referred to by the term "file," and the other three types of files are referred to by their specific names. It is important to realize, however, that they are all equivalent in the way they are accessed by Cromix through path names

Therefore, a path name is used to direct Cromix to a file, a directory, or a device. The path name usually consists of directory names separated by slashes (/), and ending in the name of the file, directory, or device desired. This serves to direct Cromix down through the tree structure described in the previous section to the file desired. Thus, the device path name

/dev/console

directs Cromix to access the console, which is a device described in the special device directory dev, which is found in the root directory (/).

In an exactly parallel manner, the file path name

/dira/diraa/fileaaa

directs Cromix to access fileaaa, which is found in diraa, which is found in dira, which is found in the root directory (/). This is an example of an **absolute path name** because it gives absolute access beginning with the root directory.

Frequently, it is more convenient to select a **current directory** containing the file desired, and then access that file directly without the necessity of a long path name. Taking our previous example, if diraa were to be selected as the current directory, then fileaaa can be accessed by the path name

fileaaa

This is an example of a **relative path name** because it gives access relative to the current directory. Any directory may be selected as the current directory.

A relative path name may be used to access levels in both the forward and reverse directions of the file system tree. The caret symbol (^) is used to indicate ancestor directory. Thus, in the previous example where the current directory is diraa, the relative path name

^filebb

could be used to access filebb in directory dira (i.e., filebb is on the same level as diraa).

To give users maximum flexibility in specifying files through path names, naming conventions of Cromix are very generous. Each path name may contain up to 128 characters and each file or directory name may contain up to 24 characters. These characters include the set of all alphanumerics (A-Z or a z and 0-9) plus several additional characters ($, __, and .).

System Security and File Protection

To be useful in a variety of everyday applications, a multi-user operating system must have several levels of **file protection** to prevent unauthorized access of files "owned" by each user. Even more important, however, is **system security,** for if the operating system itself cannot be made safe from tampering, then the file protection mechanism cannot be secured either. The Cromix operating system has several levels of both file protection and system security.

System Security

As a first level of system security, the hardware of the computer is configured to make access to Cromix very difficult. As we saw earlier, the Cromix operating system resides in its own bank of memory, and each process run by a user is assigned another bank of memory. Therefore, any user attempting to write a program which switches banks to gain access to the operating system would find that that same program would turn off the bank in which it was running!

Second, Cromix has complete password protection and password accounting. Password protection means that users will not be able to sign on to use the operating system unless they already know the correct password. Password accounting means that a log can be kept of users who sign on, when they signed on, and for how long

To oversee the operation of each

installation of Cromix, there will be at least one user with special privileges known as the **super-user.** The super-user may add or delete other users to or from the sign-on list and may change the password of any user. Likewise, the super-user is the only user who can change or delete the password accounting file.

Non-privileged users may change only their own password and have no access to the accounts file. The accounts file is an **append only** file (see section on File Protection) and thus continues to have entries added to it over time until it is emptied by the super-user.

Each installation of the Cromix operating system will determine its own system of passwords and accounts. Each password and account name may contain up to sixteen characters. To ensure that a non-privileged user does not gain access to the passwords of others, passwords are encrypted in a non-reversible code having over one trillion (10^{12}) different combinations.

As a third level of system security, the super-user may determine the sign on command line to be executed for each user. Once execution of this command line has been completed, Cromix will automatically log off that user.

For example, if users 3 and 5 are given the sign-on command line "sbasic", they will log on and automatically be running Cromemco Structured Basic. If they exit from sbasic via the "bye" command, they will immediately be logged off again. Thus, the system can be configured to allow certain users to execute only certain programs.

File Protection

Given the methods of system security described in the previous section, it is possible to ensure that only users who have been properly qualified will be permitted to use that installation of the Cromix operating system. However, once users have gained access to the system, it is also frequently desirable to limit their access to individual files and to protect those files from being either accidentally or deliberately changed.

First, all files may be created or opened by the Cromix operating system for **exclusive** or**non-exclusive access.** This means that a program which will be used to either create new files or open existing ones can be designed to perform these functions either exclusively for use by one process or not exclusively for use by that process

A file which is opened for exclusive access may not be opened by another process (i.e., by another user or another task of the same user) until it has been closed by the process which opened it originally. Likewise, a file which is opened for non-exclusive access by one process may be opened and accessed by other processes simultaneously.

This category of file protection is provided to allow the software designer using the system calls of the Cromix operating system to produce software which is free of such well-known problems as deadlock and mutual exclusion These arise when two or more users are given access to a file simultaneously, and their intended uses of that file conflict.

As a second level of file protection, a directory is "owned" by the user who originally created it, and thus the access privileges of any file in that directory may be restricted by its owner. Further, there are three possible categories on which the user may base restrictions to the access of that file. Let us examine these categories in more detail.

A user either may be the **owner** of a particular file or directory, may belong to a previously specified **group** of users having similar access privileges, or may belong to the category of all **other** users. The group to which a particular user belongs is determined by the super-user in setting up a particular installation of the Cromix operating system. Some users may not belong to any group, and others may belong to the special group of privileged users.

There are four types of **access privileges** given to a particular file in each of the just listed categories of access restriction. These are: **read access, write access, append access,** and **execute access.** A user who has one of these types of access privilege may perform only that function and no others. Usually, these privileges are assigned in meaningful combinations giving the user more than one type of access simultaneously. Thus, a user having all four types of access privilege may perform any function with a file (i.e., change it, delete it, read it, execute it, or append to it).

It is important to realize also that all four types of access privilege are assigned individually to each of the aforementioned groups of access restriction. Thus, the owner of a file usually has all four types of access privilege, but users in the owner's group or all other users may have none, one, or several of the four access privileges.

To give an example of the preceding points, the executable files sbasic, the Cromemco Structure Basic interpreter, and asmb, the Cromemco Macro Assembler, have been originally created by a Cromix system super-user. As such, the user who created them is their **owner** and has all four types of access privilege. The group to which the owner belongs is a **group** of other privileged users. However, this group has not been given the same access privileges as the owner; it has been given read access and execute access only to sbasic and asmb. This decision could be changed, of course, by the owner of the file. Finally, the owner has restricted access of all **other** users to read access and execute access only.

Thus, users who are members of the two categories "group" and "others" may execute or read the programs sbasic and asmb but will not be able to either write to them (change or delete them) or append to them (add something to the ends of them).

Many other examples can be given; however, what is important to realize is that the file protection mechanisms of the Cromix operating system are extremely versatile. Four different types of

access privileges are allowed for each file, and these four types may be assigned separately in each of three different categories of access restriction.

The Cromix Shell

The Cromix Shell is that portion of the Cromix operating system which processes and interprets all commands as they are entered at the console. Shell commands are **intrinsic** to the operating system. This means that they are executed in the system bank (see Fig. 1) of memory. **Programs** and **utilities** (i.e., operating system programs) require a separate bank of memory for execution.

The Shell performs a number of important functions for the Cromix operating system. Among its functions are to pass arguments from a typed command line to a program, to create, display, and delete files via console control; to perform "batch" processing of other Shell commands; and to handle such powerful features as sequential and detached processing and redirected I/O. Several of these features are described in more detail in the following sections.

Batch Processing

The term "batch processing" is a misleading one and one which brings to many people's minds clumsy, card-oriented automatic processing on bulky main frame systems. However, used with regards to Cromix (and previously, with CDOS), batch processing refers to the ability to have the command processor program itself, or Shell, perform an automatic sequence of Shell commands, begun by only one Shell command line. [Note that the term batch processing as used in conjunction with the Cromix operating system means sequential and not concurrent processing.]

This principle is an important one and a powerful feature of the Cromix operating system. It means that Shell commands are not required to originate at the console but instead may originate in a file whose name alone is typed on the console. Thus, complex or repetitive tasks may be performed over and over again with a minimum of typing and may also be performed by the operating system automatically.

The file name extension which is used to distinguish a file used for batch processing is cmd. This file is typable and contains the text of the command lines which are to be automatically executed. For example, suppose we create a batch file named compile.fortran.cmd containing the commands

```
for stats.stats = stats
for gauss.gauss = gauss
link stats.gauss.forlib/s.
  stats/n/e
stats
```

to automatically compile and execute a statistics program named stats.for, which requires a subroutine named gauss.for. This batch process could be executed simply by typing the command line "compile.fortran" in response to the prompt for Shell.

In addition, the batch file of the Cromix operating system may contain up to 9 parameters which are passed to it from the original command line. Thus, in the preceding example every occurrence of the name "stats" could be replaced with the expression "#1" to indicate parameter 1. Then, the command line to execute this batch process would be "compile.fortran stats", passing the required parameter from the original command line to the batch process.

Redirected I/O

One of the most powerful features available with the Cromix operating system and not found in many other operating systems for larger computers is the ability to direct input from and output to any device or file. The name given to this process is **redirected I/O**.

Redirected I/O causes the normal input of a process to be taken from and the normal output to be directed to any file or other device in the system. For example, it is possible for text normally sent to the console to be printed on the system printer by redirecting console output to the printer. Likewise, it is possible for a program normally taking its input from the console to take that input instead from another file which has been previously edited to contain the correct responses.

This process is completely general and can be set up to handle any I/O situation imaginable. There are no limitations imposed by specific devices. Whether I/O comes from a device or a file, it is treated identically by the Cromix operating system. Let us see how we can use redirected I/O.

Redirecting the normal output of a process is accomplished by using the greater than sign (>) followed by the new output file or device name on the command line. For example, the command line

```
link file1.file2.file3/m >
    /dev/prt1
```

would direct the output of the link program (in this case a map of all variables) to system printer 1 rather than to the console.

Redirecting the normal input of a process is accomplished by using the less than sign (<) followed by the new input file or device name on the command line. For example, the command line

```
testprogram < testdata3
```

would supply all input to testprogram, taking it from the file in the same directory named testdata3. Other testdata files might contain other input sequences to cause testprogram to perform other types of tests.

Many other examples of redirecting I/O could be given, from creating automatic test sequences for the computer hardware to automatically recording compilation and assembly errors in files for later review. These serve to illustrate the power and versatility of this feature of the Cromix operating system.

Sequential and Detached Processing

We have already seen how the Cromix Shell can execute more than one command sequentially by means of batch processing. However, frequently it is desirable to execute multiple processes directly from the command line. The Cromix operating system allows this in two principle ways: sequential processing and detached processing.

Sequential processing of one or more tasks is performed by speci

fying all tasks on the same command line separated by semicolons (;). The semicolon delimiter tells Cromix that the task following the semicolon is to be executed only when the task preceding the semicolon has been completed.

For example, the command line

abc ; xyz

would cause Cromix to begin and complete execution of process abc before process xyz. Task xyz will begin execution as soon as task abc is finished.

Detached processing of one or more tasks is performed by specifying all tasks on the same command line separated by ampersands (&). The ampersand delimiter tells Cromix that the task preceding the ampersand is to be executed as a detached process and that the task following the ampersand is to be executed as a **concurrent process.** In other words, these two processes are to begin execution simultaneously.

A detached process is identified by a **process identification number,** or **PID,** which is displayed on the terminal when the process begins execution. This process requires an additional bank of memory beyond that required for the concurrent process (see Fig. 1). To give an example, suppose we type the command line

abc & xyz

This tells Cromix to begin task abc as a detached process and to execute task xyz as the concurrent process. The Cromix Shell will not again prompt the user until process xyz has been completed regardless of the status of process abc. Two banks of memory (in addition to the bank required for Cromix itself) are required by this command line, one for task abc and one for task xyz.

Now suppose we type the command line

abc & xyz &

This tells Cromix to begin both tasks abc and xyz as detached processes. Thus, the Cromix Shell will prompt the user again immediately. Two banks of memory are required by this command line; however, if the user were to start a concurrent process once the prompt had been received, three banks of memory would be required.

It is often useful to execute detached processing and redirected I/O together. While a task is being run detached, its output can be redirected to a file or device other than the console. Its input should also be redirected from a file. If it is not, Cromix automatically directs input from the **null** device. This leaves the console free for running a concurrent task.

Finally, in simultaneously using the two types of processing just described, it is sometimes necessary to force the operating system to wait until all detached processes have finished execution. This is the purpose of the **wait** intrinsic command. For example, the command line

abc & def ; wait ; pqr & xyz

tells Cromix to execute processes abc and def concurrently, to wait until they are both finished, and then to execute processes pqr and xyz concurrently.

Note the way the Cromix operating system makes the most efficient use of system memory in executing detached processes. If one or more banks of memory are not currently being used by other users of the system, they are then free for running detached jobs. Once the detached jobs are finished, the memory will again be free for other users.

Other Features

This section describes several other outstanding features of the Cromix operating system which contribute to its versatility and ease of use.

System Time and Date

A required feature of any multiuser, multitasking operating system is the ability to keep track of the time and date. These are then made available to any process which requires them.

The Cromix operating system maintains the time and date and makes these available for several important tasks. First, the time that each user logs on and logs off the system is stored in both the **account file** and the **who file.** The who file is accessible by any user of the system to determine who is signed on at any given time. The account file is for use primarily by the super user, and is used to compute job or user cost accounting of system time used. Appropriate data is automatically appended to this file each time a user logs on or logs off the system.

Second, Cromix automatically stores the current time and date with any file which is created or updated. This makes it possible for users writing software or documentation to easily keep track of their latest revisions of files. The operating system performs the overhead automatically.

Finally, the system time and date are made available to any user process or program which requires them.

Speed of Operation

All operating systems require a certain amount of overhead time to perform their functions, particularly in calculating the locations of files through their file management systems. Since this overhead time varies considerably from one operating system to the next, it is an important parameter to be examined for each operating system.

The Cromix operating system is structured around a very powerful file system as we have seen in previous sections. The key information describing each file (such as its size, when it was created and by whom, and what file protection attributes it has) is stored together in a block near the beginning of the mass storage device containing the file. This block of information is called an **inode.** Each inode is numbered and can be referenced through its number.

Each directory containing that file has a second, smaller block of information which contains the file name and its inode number. Thus, multiple directory entries can refer to the same file with no degradation of system speed. Because the inodes are numbered and in known locations, they may be referenced immediately and without accessing any file directories. This provides the Cromix operating system with the fastest possible file access speed.

As we have seen, the Cromix operating system resides in its own bank of memory (see Fig. 1). A large portion of this memory is

reserved for high-speed data I/O buffers for transfers to and from mass storage devices such as disks and character devices such as terminals and printers. The disk buffers are allocated such that reads and writes to a disk are kept to an absolute minimum. A large sector size (512 bytes) is used to further minimize reads and writes during sequential access.

The large number of I/O buffers was designed into the Cromix operating system to complement and take full advantage of the high file access speeds already possible with its file management system. These two factors combine to give Cromix one of the highest speeds of operation of any operating system implemented on any microcomputer.

Executing CDOS Programs Under Cromix

A concern of many prospective users of Cromix is its ability to run present Cromemco software designed for the CDOS operating system. Cromix is an entirely new operating system which in no respect resembles CDOS internally. It has its own system calls and its own I/O configuration. However, in the interests of maintaining compatibility with present software designed for use with CDOS, Cromemco has designed a **CDOS Simulator** program to run under the Cromix operating system. This program intercepts CDOS system calls and translates them into Cromix system calls; thus, it "simulates" the operation of CDOS under Cromix.

One concern many users have initially about using the CDOS Simulator is whether it will execute system calls at a slower speed than when running directly under the CDOS operating system. However, this is not the case. The operation of the Cromix operating system is so significantly faster than that of CDOS, particularly for file access, that the effective throughput rate of running a CDOS program under the CDOS Simulator using the Cromix operating system is higher than running it directly under the CDOS operating system!

From the preceding it is apparent that two types of executable files are permitted to run under the Cromix operating system. These are distinguished by their file name extension (i.e., the portion of the file name which follows the last embedded period (.). Programs which are directly executable under Cromix using Cromix system calls have the file name extension of **bin**, which originally derived from the word "binary".

Programs which are executable under Cromix by means of the CDOS Simulator, which simulates the desired CDOS system calls, have the file name extension of **com**, which originally derived from the word "command" and matches the file name extension of these programs under CDOS.

These two types of executable files are kept together in the same directory, named the **bin directory**. When executing com files, Cromix will automatically first load the CDOS Simulator. Therefore, actual execution of either of these two types of files appears identical to the user of Cromix. This is a very powerful feature of the Cromix operating system and guarantees that present software is not made obsolete.

Conclusion

We have seen that the functions of an operating system for a computer are: to supervise the execution of other programs, to perform all disk and device I/O, to manage the computer's primary memory, and to allocate CPU time. We have examined these features as they are implemented in the Cromemco Cromix Operating System.

The Cromix operating system is a multi-user, multi-tasking operating system handling up to six users or processes simultaneously. Cromix is distinguished by an elegant and very fast file management system characterized by a tree data structure of hierarchical directories. Access of any file or device in the file system is accomplished by traversing one of the branches of this tree.

The Cromix operating system has many levels of both system security and file protection. A particular system installation may be protected from unauthorized use by both password protection and user accounting. Files and directories are protected by four types of access privileges, and these types may be applied separately in each of three categories of access restriction. Files may also be accessed either exclusively or non-exclusively by a program.

The Cromix operating system can perform multiple processes for each user from one command line. One or more tasks may be detached from the terminal and performed in the background. When this is done, Cromix can redirect all I/O for that process to and from disk files. I/O from/to any device may be redirected from/to any other file or device in the file system. Cromix also can perform sequential processing of tasks, either from the command line or, automatically, through batch processing.

To maintain compatibility with older software, the Cromix operating system includes a CDOS Simulator which allows CDOS programs to be executed directly under Cromix. The high speed of operation which Cromix offers allows these programs to run even faster under Cromix than they do under CDOS.

These features and others combine to make the Cromemco Cromix Operating System one of the most powerful operating systems yet implemented on any microcomputer.

Bits & Bytes, Nibbles & Tweaks

RAFTOR USERS' GROUP

A RAFTOR users' group is based at the Berkeley-Livermore Laboratory. For more information, contact: Skip Egdorf, Box 735, Los Alamos, NM 87544.

MEMBER NEEDS SOFTWARE/HAS SOFTWARE

Jim Knowles, one of the first to join the IACU, needs a loan package that will work for a federal credit union (he indicates that a federal savings & loan package might do the job). Should include provisions for members' deposits, loans, payments, interest (both ways), and current balances.

Jim also has a couple of interesting packages available: one for Auction Sales (Point of Sale application) in SBASIC; the other for Social Service Agencies (with flexible Fiscal Year application) in 16K Extended BASIC. Contact Jim at: (312) 695-7775.

Alan Grayson of Charlotte, North Caroline is interested in getting together with other users in his neck of the woods (this may be the beginning of a local users' group) to discuss common interests. Alan is also looking for engineering software, especially dealing with Stress Analysis, Project Engineering, and anything that will help him cost projects. He can be reached at (704) 554-2043.

Clarence Laney in San Jose, California is looking for some tips for converting his system from CDOS 1.07 to CDOS 2.17. His system consists of a Z-2 with 64K memory, SOROC IQ-120, PRINTERM 879 printer, and two SHUGART 801 Disk drives. And therein may lie the root of the problem, because SHUGART and PERSCI drives operate at different speeds. So, if anyone out there has made this kind of conversion successfully — and we do stress successfully — let us know. We'll pass it along to Clarence, and also store the information for the next time the problem crops up.

CROMEMCO USERS' GROUP IN NORTHERN CALIFORNIA

Active users' group in Walnut Creek, meets bi-monthly. For details, contact: Hank Couden at (415) 935-6502. Meetings held at:
COMPUTERLAND OF WALNUT CREEK
1815 Ygnacio Valley Road
Walnut Creek, California 94596

SOFTWARE NEEDED

Jerry Morlock in Bellingham, Washington is looking for some software. He needs a Federal Income Tax preparation program that will print forms 1040s; a complete accounting package, including payroll, that will run on Cromix; and a job costing package for a General Contractor. Contact us, we'll pass all information along to Jerry. Better yet, if you have such packages, start advertising in I/O News.

WANT TO START A LOCAL IACU CHAPTER?

We'll help you publicize your group, and we have a few helpful hints to pass along. Write IACU.

DO YOU HAVE AN ESTABLISHED LOCAL USERS' GROUP?

Let us know about it — when and where you meet, who to contact, etc. We'll put the data on file to advise other members who ask.

SOFTWARE SEARCH

Early in July we mailed inquiry letters to over one hundred listed software houses. We have received fewer than 20 replies. We are trying to build an active and viable Software Data Base, but we need help. BOy, do we need help! This is a straightforward matter of supply and demand, and our members have indicated one heck of a demand. So, if you know any suppliers of quality software — packages that work like magic on Cromemco systems — tell them we would really like to hear from them. Then, you'll hear from us.

WHERE IS YOUR MEMBERSHIP CARD?

It's on its way. We delayed ordering the numbered Membership cards until we had a feeling for the quantity we would need. You will begin receiving your cards shortly after you receive this issue. Three ring binders, big enough to hold Volumes I & II of I/O News will arrive shortly thereafter.

PRESIDENT NIXON JOINS IACU

A membership application for Richard E. Nixon, President of Nixon's Truck Parts, Inc. in New York, was received in late August. His application was the 700th received since the association was formed in May.

SPECIAL MEMBERS

Special Memberships are open to authorized Dealers & OEMs only. These memberships cost $350 per year, and entitle the member to a special listing on the Association's Referral Service Data Base, as well as this printed listing.

ALASKA SYSTEMS
2900 W. Northern Lights Blvd.
Anchorage, AK 99503
(907) 248-2700

Full service company, providing consultation. Stocks a large inventory of both hardware and software; specializes in payrolls, power generation monitor, custom device drivers. Provides warranty repair service.

Key Personnel: William Hutchison, Owner
R. Pereira, software specialist
R. Yorgenson, software specialist

Major Market Area:
Sales & Service: throughout Alaska

COLLINGSWOOD COMPUTER CENTER
684 Haddon Avenue
Collingswood, NJ 08108
(609) 854-1333

Medium-sized software house, specializing in small business systems; all models of Cromemco/payroll, billing, mass mailer. Provides warranty service also contract and hourly service.

Key Personnel: Jim Lenz, Pres. (software design & development)
Carol Bates, Sales Dir. (Sales & marketing)
Jim Masterson, Vice Pres. (service)

Major Market Area: Sales: New York to Washington, extending to entire U.S. Service: Metro Philadelphia extending to Eastern Corridor.

COMPU'DEV INT'L
4750 Von Karman
Newport Beach, CA 92660
(714) 551-0416

Key Personnel: Frank J. Marshall, Ph.D., Pres. (hardware & software consultant)
Paula Joyner, Gen. Mgr. (senior programmer)
Kathie Butler, Mktg. Rep. (sales)

Major Market Area: Sales: International.
Service: National.

COMPUTER CENTERS OF AMERICA
2129 Westheimer Road
Houston, TX 77098
(713) 527-8008

Complete line of hardware and software in inventory, 10,000 square feet of space. Provides warranty repair service.

Key Personnel: Avery More, Pres. (sales)
Richard Hemeard, Dallas Mgr.
Race Feirman, Houston Mgr.

Major Market Area: Sales & Service: Houston and Dallas, Texas.

COMPUTERLAND OF ITHACA
225 Elmira Road
Ithaca, N.Y. 14850
(607) 277-4889

Two stores; a computer store on 225 Elmira Road, a software house on 609 W. Clinton Street. Carries a broad line of commercial application software using Cromemco 16K BASIC. Stocks other Cromemco products. Provides custom programming and consulting, as well as warranty service.

Key Personnel: H.G. Vaughan, Pres. (software development, consulting)
P.C. Vaughan, Vice Pres. (technical sales, retail)

Major Market Area: Sales: Central New York, extending world-wide. Service: Central New York.

COMPUTER PRODUCTS INTERNATIONAL
3225 Danny Pkwy.
Metairie, LA 7002
(504) 455-5330

Software house, specializing in general business and wholesale distributor systems. Full line of Cromemcos, in 10,000 sq. ft. two story building.

Key Personnel: Dan Ellis, Pres.
Jim Miller, Vice Pres. (systems)
Ben Rauch, Vice Pres. (systems)

Major Market Area: Sales: Entire Gulf Coast. Extended area, U.S.

CUSTOM COMPUTER SPECIALISTS, INC.
208 Roanoke Avenue
Riverhead, NY 11952
(516) 396-2199

Systems house with retail showroom. Stocks a full inventory of Cromemco hardware and software. Provides warranty service and consultation.

Key Personnel: Gregory G. Galdi, Pres.

Major Market Area: Sales: Northeast U.S., extending to East Coast. Service: East Coast extending to Continental U.S.

DIGITUS LIMITED
9 Macklin Street
Coven Garden, London WC2
01-4056761 Telex - Ref: 3005

Complete line of Cromemco hardware and software, 4 000 sq ft of space Specializing in training, management sciences, personnel systems. Provides consultation services.

Key Personnel: A.C. Wood, Managing Dir. (consultancy)
P.S. Woolfonden, Systems Mgr. (software)
S. Parel, Sales Mgr. (hardware)

Major Market Area: Sales & Service: London, extending throughout United Kingdom.

INFORMATIVE SYSTEMS P/L
3 Bank Street
South Melbourne, Victoria, Australia
03-6902284 TWX 30458

Full range of Cromemco, retail and wholesale computer store. Provides full sales and service, specializing in education and small business applications.

Key Personnel: N. Rosenbaum, Director (sales)
B. Bryan, Director (technical)
S. Rosenbaum, Director (software)

Major Market Area: Sales & Service: throughout Australia

INFOSOFT SYSTEMS, INC.
25 Sylvan Road South
Westport, CT 06880
(203) 226-8937

Supplier of sophisticated software to systems users and retailers. Complete line of Cromemco software and applications packages in stock, as well as hardware. Special interconnects to accommodate Cromemco software.

Key Personnel: Ken Short, Pres. (sr. programmer/analyst)
Richard Roth, Vice Pres. Mktg. & Production
Peggy Herlihy, Customer Service (analyst)

Major Market Area: Sales: International. Service: U.S.

MICAH APPLICATIONS & HARDWARE
73 Cazneau
Sausalito, CA 94965
(714) 332-4443

Software development and consulting house. Specializing in education, resorts, construction, and home health services. Special financing available for software development projects.

Key Personnel: Ninad Freedman, Pres. (software engineer)
Dale Maxwell, Sales Mgr.

Major Market Area: Sales: International. Service: Bay area & Western U.S.

MCM ENTERPRISES
459 Hamilton Ave., #304
Palo Alto, CA 94301
(415) 493-3333

A full service computer solutions company with consulting, equipment, software, training, and service. MCM carries a full line of Cromemco Systems, Serendipity and Computer Information Systems software, and NEC Printing Terminals. Authorized NEC Service Center for Northern California (printers & printing terminals).

Key Personnel: M.C. Merchant, Owner (systems design)
G. Nielsen, Svc. Engr. (maintenance)
L. Yori, Mgr., Reno Office (systems design)

Major Market Area: Sales: San Francisco Peninsula & Nevada extending internationally. Service: S.F. Peninsula, Nevada, extending into Northern Calif.

Reno Office: 1275 Kleppe Lane, #14, Sparks, Nevada 89431 (702) 358-0415

TRADEWIND SYSTEMS
Box 96
Liberal, KS 67901
(316) 624-8111, O/S KS 1-800-835-2057

Exclusive Cromemco dealer, specializing in complete business systems. Provides consulting services. Full inventory.

Key Personnel: Clark Stewart, Pres. (business systems)
Wayne Stewart, Vice Pres. (tech./software)
Kevin Elmore, Programmer/analysis

Major Market Area: Sales: S.W. Kansas, extending to COlorado, Kansas, Oklahoma, Texas, New Mexico. Service: S.W. Kansas.

XITAN SYSTEMS, LTD.
23 Cumberland Place
Southampton, England
0703 38740

Complete line of computers, 1,000 sq. ft. Also add-ons for Cromemco. All Cromemco software included in inventory. Provides warranty repair service.

Key Personnel: G.C. Lynch, Managing Dire. (sales & technical support)
R. Wilmott, Product Support Engineer (engineering)
J. Rosser, Personal Assistant (order processing)

Major Market Area: Providing U.K. & E.E.C. with total business, educational scientific, medical and industrial support.

Join Now. But Don't Cut The Page. Copy This Application and Mail to IACU With Your Check.

Application for Membership

Country ______________________ Zip ____________

Phon ______ ____________ ______
area code number

Mail Your Membership Application to:
The International Association of Cromemco Users
P.O. Box 17658, Irvine, CA 92713
For more information, call (714) 955-0432

The International Association of Cromemco Users is designed to provide its Members with the information they want. Help us deliver by answering the following questions. You may check more than one block as applicable: My field is:

Educational Institution ______________________
(indicate level)

My Primary Uses are:

( ) Personal Only
( ) Process Control
Other ______________

I am Interested in:

( ) Software
( ) Other ______________

I Want to Know More About the following Packages:

( ) Accounting
( ) Home Economics
( ) Medical Research
( ) Process Control
( ) Sports & Games
( ) Other ______________

Describe Your Present System:
(use brands and model numbers)

List Software Now in Use:
(Packages, Special Operating Systems, etc.)

What Types of Software Would You Like to know more about?

Would you be interested in preparing an article of interest to our members?

The CDOS* Active Command File

© Copyright 1980
by Darwin A. Engwer
Dynamic Systems Group

Darwin Engwer is the president of Dynamic Systems Group, Calgary, Alberta. His firm serves clients in Canada and the U.S., specializing in microcomputer system configuration planning and subsequent program generation. Darwin graduated with honors from the Electronic Engineering program at the Southern Alberta Institute of Technology and has been creating microprocessor based control systems for Alberta industries for the past three years.

Ever wonder exactly how the CDOS active command file ($$$.CMD) works and how you can use it to have an application program initiate a CDOS command sequence and then return to the application program (for example, performing a file backup sequence for an end user — under control of a BASIC program) ? If 'Yes' then read on, else stop!

The active command file (ACF) is normally created by the batch (@) utility program from information contained in a user command file or from commands entered from the keyboard when the one time mode is used. This file has the name $$$$.CMD (or $$$$n.CMD under a multi-user version 1.5 system, where n is the user's number) and is checked for by CDOS each time it is 'warm started'. If the file exists, CDOS reads the next command line from it instead of the terminal's keyboard.

File Organization

ACF records are 128 bytes long — the record size used by all applicable CDOS routines. The most important fact to note about the ACF organization however, is that the pending commands are stored in reverse order, i.e., the next command is always contained in the last record of the file. The reason the commands are stored in reverse order is so that the next command can be easily 'removed' from the file by decrementing the record count (and 'de-allocating' a cluster every eighth record) in the directory entry (see FCB format on page 72 of the CDOS manual). If the commands were stored sequentially in the ACF all the remaining commands would have to be shifted up one record each time a command was removed — a tedious process. This organization is also the factor which limits the maximum number of pending commands to 128 (see page 48 of the CDOS manual), since 128 records * 128 bytes/record = 16384 bytes (one full extent).

Record Format

The first byte of each ACF record is the length (in binary) of the command string contained in the record. This byte count is required in order for the string to meet the requirements of a standard command line when it is loaded into the command line buffer (see page 90 of the CDOS manual).

The next field in the record is the ASCII representation of the command string. The maximum length of the string is 126 bytes (125 when input using the one time mode of BATCH) although it is normally substantially shorter.

The last field in an ACF record is one byte long and immediately follows the command string. It consists of the ASCII character '$'. After the command string is loaded into the command line buffer, CDOS 'echoes' it to the console using the 'print line' call (#9, described on page 68 of the CDOS manual). The dollar sign is the string terminator for this call. Afterwards it is changed to a binary zero in order to conform to the standard command line format.

Example Application

Now that we know what the active command file looks like and a little bit about how it works, let's investigate an example in 32K SBASIC that performs the file backup operation mentioned above.

```
10  CREATE "$$$$.CMD"
20  OPEN \1,128 \"$$$$.CMD"
30  DIM C$(125)
40  C$ = "XFER/C C:DATAFILE.BAK = B:DATAFILE.
      DAT"
50  PUT \1,0\ CHR$(14);"BASIC MENU.SAV"; '$'
60  PUT \1,1\ CHR$(LEN(C$));C$(0,LEN(C$)-1);'$'
70  CLOSE \1 \
80  BYE
```

The function of lines 10 and 20 is to create and open the active command file. This could be made more general by first checking to see if the file already exists and if it does, read records until an EOF error occurs, then add your commands in successive records.

Lines 30 and 40 define the command string necessary to perform the file backup operation, using XFER.

Next two records are written to the ACF (lines 50 and 60) in the reverse order to which they are to be executed. Notice that carriage returns and/or line feeds are not required in the ACF.

The file is then closed and BASIC is terminated via a BYE command (lines 70 and 80).

The last operation causes a CDOS warm start, whereupon the ACF presence is detected and the last record is read into the command line buffer. This causes the file B:DATAFILE.DAT to be backed-up onto drive C under the name DATAFILE.BAK. The next ACF record causes BASIC to be reloaded, running the program MENU.SAV.

(Editor's Note: We tried it, and we like it. The application of this is limited only to the user's imagination. Although the example is in BASIC, the same logic can be applied to other languages. We liked this so well we asked Darwin Engwer to send us more. We hope other Members will supply articles just as concise and helpful.)

*CDOS is a trademark of Cromemco Inc., Mountain View, California.

Computer Generated Graphics Made Easy

— Dr. David McLennan*

A computer generated graphics system is only as good as the support software available for that system. The problem w[illegible] graphics systems is [illegible], that part or all of the programming must be written in machine language code. Generating graphics displays under this condition becomes a very tedious and time consuming operati[illegible] Cromemco's new graphics software package (SGS L or SGS S) guarantees that this will not be a problem for those using its SDI graphics interface. This package gives the user a choice of programming languages and as a result the job of generating video graphics displays becomes a simple, straightforward operation.

[illegible] languages available include Cromemco's Macro Assembler, Cromemco's Extended BASIC, Cromemco's Structured BASIC, Cromemco's FORTRAN IV, and Cromemco's Rational FORTRAN. With the exception of Extended BASIC, all of these available programming languages make it possible to use graphics calls which suggest the operation being performed. For example, the call

[illegible]

will draw a solid rectangle with the lower left corner at the coordinate x1,y1 and the upper right corner at the coordinate at x2,y2. The parameter c has a value between 0 and 15 indicating the location in the color map corresponding to the color to be displayed. Similarly the call

XLINE (x1,y1,x[illegible]

would dr[illegible] two points (x[illegible] the designate[illegible]

The graph[illegible] [illegible] CDOS comma[illegible] [illegible] saved to (or lo[illegible] [illegible] These com[illegible] [illegible] a number of attributes can be used with each of these commands. A command file for generating user defined color maps is also included in the package. This command (CMAPGEN) actually displays the color map on the monitor and continually updates it as changes are made. The only other software required for the use of the high [illegible]solution graphics package is one of the programming languages.

[illegible] [illegible]ware required for full [illegible] of this package consists of a [illegible]emco Z 80 based computer [illegible]n with floppy disk, 64K of [illegible] Access Memory (RAM), [illegible]DI graphics interface boards, a three gun color monitor, and two 48K Two Port (48KTP) [illegible]ry boards. One of the TP memory cards is addressed for bank 5 and the other for bank 6 [illegible]ming purposes they [illegible] referred to as display pages 0 [illegible]pectively. Although the system will operate with only one 48KTP memory card it is not recommended since features such as windowing, rotation, and animation are lost.

The following examples demonstrate the ease with which [illegible]aphics can be generated using [illegible]s package. In the first example Structured BASIC will be used for the programming language while [illegible] second example, Rational [illegible] will be used. Suppose [illegible] w a large bull's eye [illegible] filled [illegible] different co[illegible] [illegible]ed circle [illegible] component will vary between 1 and 241. The x and y components for the screen center will therefore be x = 189 and y = 120 as shown in the following figure. ⟶

A program which would achieve our objective is:

```
 10 INIT
120 FOR C = 15 TO 0 STEP -1
130 XFCIR (189,120,C*7,C)
140 NEXT C
500 END
```

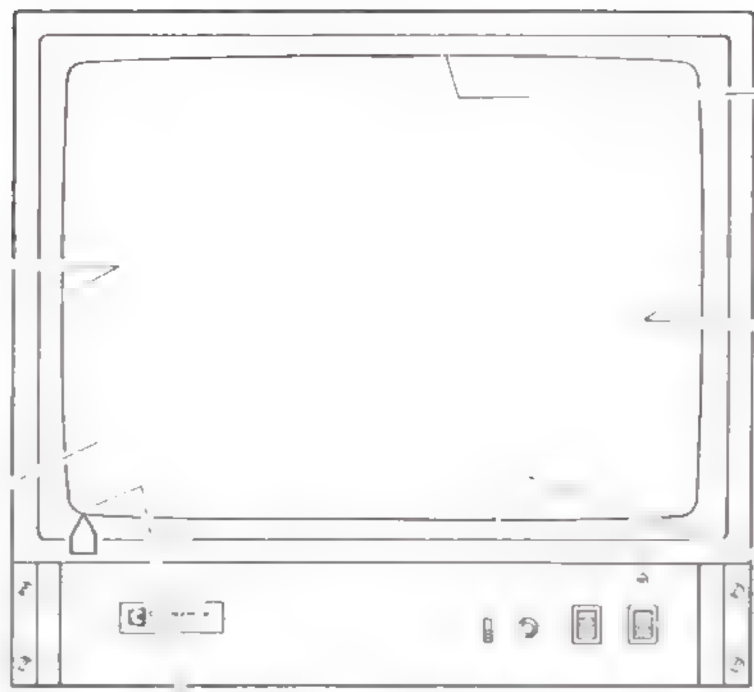
A few comments regarding line 10 of the above program are in order. The call INIT would normally be the first graphics call in a prog[illegible]m written in SBASIC and would perform the following functions:

- turn on the SDI
- clear page 0 and display it
- make page 0 the work page
- reset the color map to the standard preset values
- [illegible]n[illegible]ate the function CLIP
- set scaling to the default values
- select the 16 color medium resolution display mode

Suppose that after completing the bull's eye we wish to cycle the colors of the bull's eye through all 4096 possible values. The following instructions could then be added to the above program to achieve this result.

[illegible]

*Dr. McLennan is Technical Training Manager at Cromemco Inc.



The above figure shows the coordinate system used by the Cromemco graphics software package for operation both in the medium resolution and the high resolution mode. In the medium resolution mode the x coordinate varies from 1 to 378 when going from left to right across the screen. The y coordinate goes from 1 to 241 when going from bottom to top

page displaying the graphics contained on a second page. In this case the graphics on the second page is written text

```
 20 DIM T$(30)
 30 T$ = "Have a good day
 40  WORKON (1)
```

The call .WORKON (0 or 1) designates which page subsequent graphics instruction will effect

```
 45 .XAREA (50.50.400,200,12)
```

This line of code generates a colored background for the text

```
 50  XTEXT (120,115,10,T$)
```

The format of the above call is XTEXT (x,y,c,text)

```
 60  WORKON (0)
 150  WINIT (0)
```

This call specifies that for any windowing calls page 0 is to be displayed and that designated window openings are for that page

```
 160  WOPEN (8,110,17,130)
```

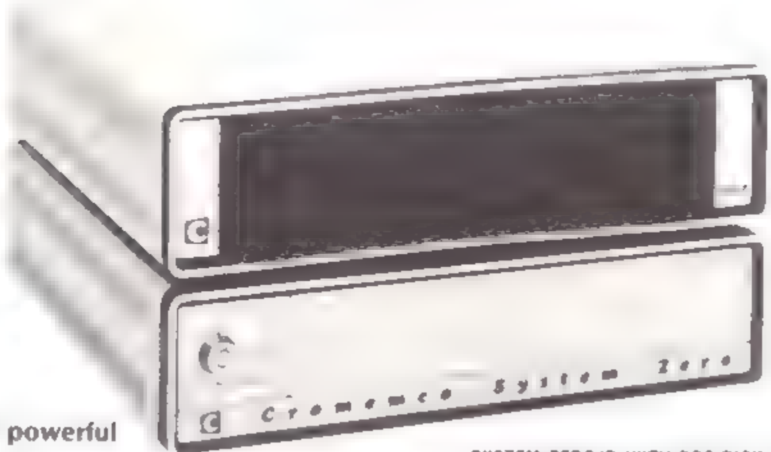
When the above lines are added to our program a bull s eye is drawn using the normal ... map. A window is then ope... the bull s eye displaying the phrase Have a good day in a bright orange color with a black background. The program will begin cycling the colors of the bull s eye and in the proc... color of the phrase and its background will change periodically.

The next example is a RATFOR program and illustrates the use of some of the graphics calls in both the high resolution and the medium resolution mode. The GRAFIX call moves the FORTRAN stack and therefore must be the first call in any FORTRAN or RATFOR program. Note that in the following code the letter H is added to a call to designate high resolution

```
# Program : MODCMP
# Purpose : To compare hi res
and med res mode

    call        grafix
    call        init
# Redefine color map
    call        defclr(10,15,15,0)
    call        defclr(11,15,10,0)
    call        defclr(12,10,15,0)
    call        defclr(13,0,4,15)
    call        defclr(15,0,15,4)
# Med res mode
    call        xcirc(40,210,20,9
                xline(50,150,150,230,10)

    call        xarea(240,190,280,220,
 2)
    call        xtext(100,150,13, MED
RES mode )

 # Hi-res mode
    call      resbox(1,1,24,120)
    call      hxcirc(40,90,20)
    call      hxline(50,30,150,110)
    call      hxfcir(200 90 25)
    call      hxarea(240,70,280,100)
    call      hxtext(100,30, HI-RES
mode")
END
```

The above examples have only touched the surface in demonstrating the ease and tremendous flexibility of Cromemco's high resolution graphics package. Only by visiting your Cromemco dealer and trying the system yourself, will you be convinced how easy and enjoyable it is to create your own computer generated graphics.

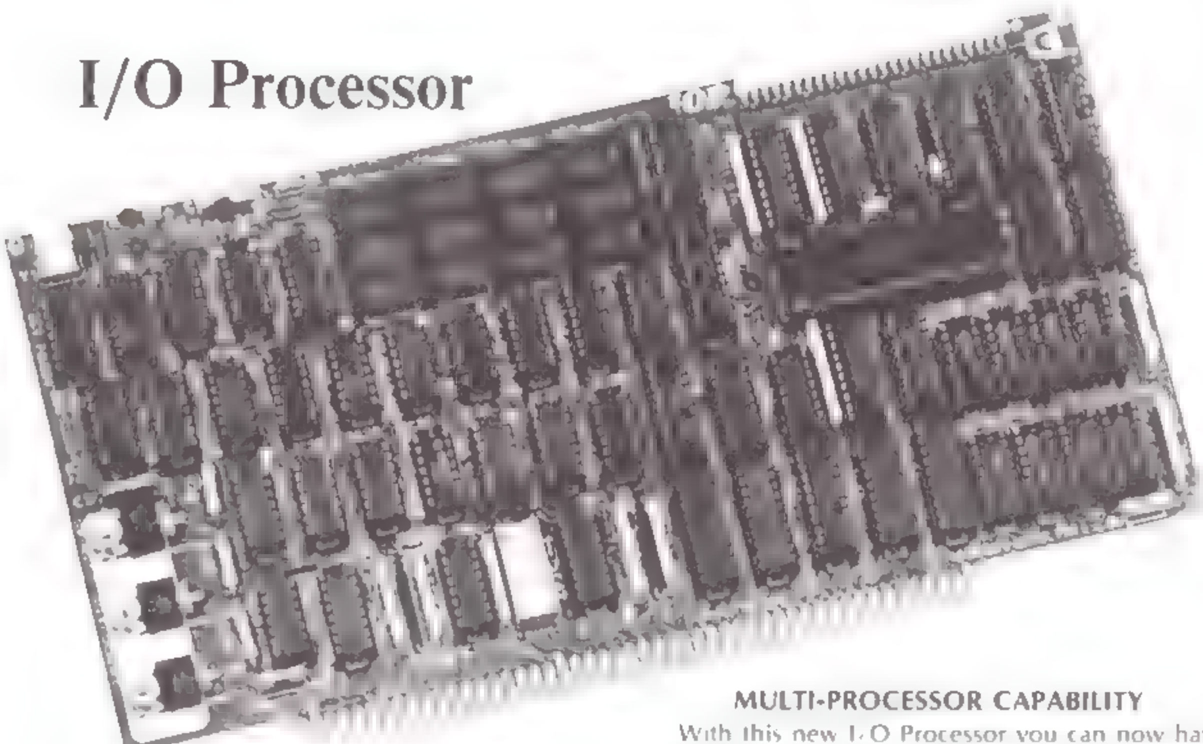
System Zero Computer



SYSTEM ZERO/D WITH DDF DISK DRIVE

- Small but powerful
- Up to 780K bytes of diskette storage
- Self-test diagnostics

I/O Processor



MULTI-PROCESSOR CAPABILITY

With this new I/O Processor you can now have multi-processor capability in your S-100 bus system.

The new Model IOP is a true single card computer — one that has a fast Z-80A processor, 16K bytes of RAM, and up to 16K bytes of PROM capacity.

The IOP interfaces to the S-100 host processor by means of simple "input" and "output" instructions.

High-Performance Color Graphics System

- A professional color system
- High resolution
- A powerful computer
- NTSC conformance
- A low price

See Your Company In I/O News

Display Advertising Rates

Front Cover (4-color)	$1,100
Inside Front Cover, Back Cover, and Inside Back Cover (4-color)	950
Full Page	650
2/3 Page	500
1/2 Page	350
1/3 Page	250
1/4 Page	190
1/6 Page	135
1/12 Page	70

For two-color ads (black & one process color) add $200 to the rates listed above.

ASI INTEGRATED BUSINESS SOFTWARE

****

RECEIVABLES	ORDER ENTRY
PAYABLES	INVENTORY
PAYROLL	JOB COSTING
GENERAL LEDGER	

****

PLUS TECHNICAL SUPPORT
FOR END USERS AND DISTRIBUTORS

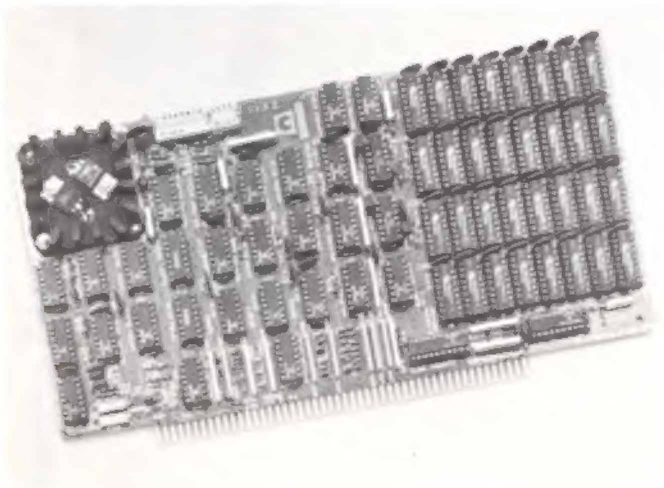
Available for 8080/Z80 CP/M systems on 8" diskettes and Northstar compatible 5¼" diskettes. Runs on Cromenco-Z2H hard disk system. CDOS compatible.

ARKANSAS SYSTEMS INC.

8901 Kanis Rd.
Suite 206
Little Rock, AR 72205
(501) 227-8471

Price Reduced On 16KZ RAM Card

Cromemco recently announced a reduction in price for their popular 16KZ RAM memory card. The list price of the 16KZ RAM card has been reduced from $595 to $495.

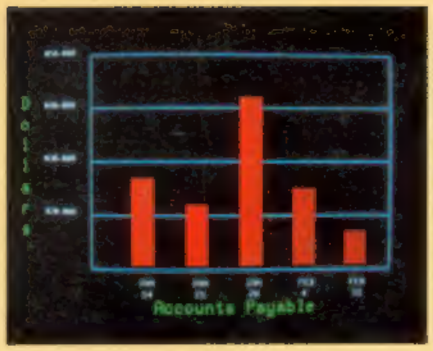


The 16KZ features S-100 compatibility, fast operation with no wait at either 4 MHz or 2 MHz, expandability to a half megabyte with Bank Select, Direct Memory Access Override Circuitry, and time-sharing capabilities.

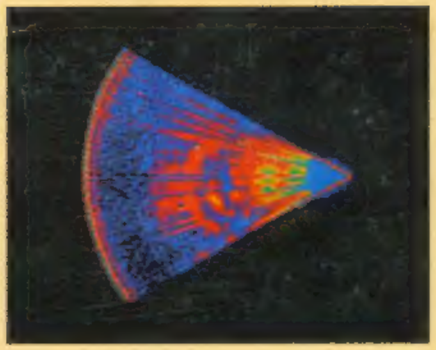
Bank select is an especially powerful feature which provides a convenient method by which to expand system memory space beyond 64K and also allows you to achieve a time-sharing system with a minimum of software overhead. With bank select, memory space may be organized into 8 banks of 64K bytes each for a total of one-half megabyte of memory. Consequently, for time-sharing systems, up to seven users can use the system simultaneously with each confined to his own bank of memory.

The Direct Memory Access Override Circuitry allows any DMA device to access different banks of memory during DMA operations. This capability permits DMA devices, such as video graphic display generators to remain "in contact" with their own dedicated memory area regardless of the current memory bank selection.

The 16KZ RAM card (Model 16KZ) is available for immediate delivery.



Management Information Display



Ultrasonic heart sector scan

High-resolution display with alphanumerics

Get the professional color display that has BASIC/FORTRAN simplicity

LOW-PRICED, TOO

Here's a color display that has everything: professional-level resolution, enormous color range, easy software, NTSC conformance, and low price.

Basically, this new Cromemco Model SDI* is a two-board interface that plugs into any Cromemco computer.

The SDI then maps computer display memory content onto a convenient color monitor to give high-quality, high-resolution displays (756 H x 482 V pixels).

When we say the SDI results in a high-quality professional display, we mean **you can't get higher resolution than this system offers in an** NTSC-conforming display.

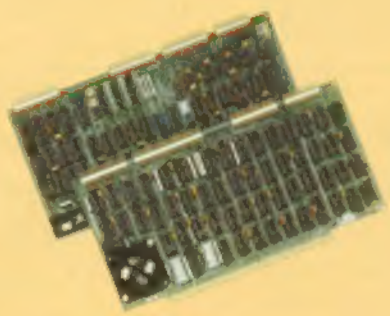
The resolution surpasses that of a color TV picture.

BASIC/FORTRAN programming

Besides its high resolution and low price, the new SDI lets you control with optional Cromemco software packages that use simple BASIC- and FORTRAN-like commands.

Pick any of 16 colors (from a 4096-color palette) with instructions like DEFCLR (c, R, G, B). Or obtain a circle of specified size, location, and color with XCIRC (x, y, r, c).

*U.S. Pat. No. 4121283



Model SDI High-Resolution Color Graphics Interface

HIGH RESOLUTION

The SDI's high resolution gives a professional-quality display that strictly meets NTSC requirements. You get 756 pixels on every visible line of the NTSC standard display of 482 image lines. Vertical line spacing is 1 pixel.

To achieve the high-quality display, a separate output signal is produced for each of the three component colors (red, green, blue). This yields a sharper image than is possible using an NTSC-composite video signal and color TV set. Full image quality is readily realized with our high-quality RGB Monitor or any conventional red/green/blue monitor common in TV work.

Model SDI plugs into Z-2H 11-megabyte hard disk computer or any Cromemco computer

DISPLAY MEMORY

Along with the SDI we also offer an optional fast and novel **two-port** memory that gives independent high-speed access to the computer memory. The two-port memory stores one full display, permitting fast computer operation even during display.

CONTACT YOUR REP NOW

The Model SDI has been used in scientific work, engineering, business, TV, color graphics, and other areas. It's a good example of how Cromemco keeps computers in the field up to date, since it turns any Cromemco computer into an up-to-date color display computer.

The SDI has still more features that you should be informed about. So contact your Cromemco representative now and see all that the SDI will do for you.

Cromemco
incorporated
280 BERNARDO AVE., MOUNTAIN VIEW, CA 94040 • (415) 964-7400
Tomorrow's computers today